Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

Пермский национальный исследовательский политехнический университет

(ПНИПУ)

Факультет: Электротехнический (ЭТФ)

Направление: 09.03.04 – Программная инженерия (ПИ)

Профиль: Разработка программно-информационных систем (РИС)

Кафедра информационных технологий и автоматизированных систем (ИТАС)

Допустить к защите

Зав. кафедрой ИТАС: д-р экон. наук, проф.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р.А. Файзрахманов

«\_\_\_\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

на соискание академической степени бакалавра

на тему

**«****Разработка программного обеспечения задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM»**

|  |  |  |
| --- | --- | --- |
| Студент: |  | Савин Сергей Алексеевич |
|  | (подпись, дата) |  |
| Группа: | РИС-18-1б | |

Состав ВКР:

1. Пояснительная записка на 109 стр.
2. Приложения на 5 стр.

|  |  |  |
| --- | --- | --- |
| Руководитель ВКР: |  | канд. техн. наук, доцент каф. ИТАС  О.А. Полякова |
| (подпись, дата) |
| Консультант по предметной области: |  | руководитель проектной группы  А. А. Бартошевич |
| (подпись, дата) |
| Прохождение предзащиты: |  | канд. техн. наук, доцент каф. ИТАС  О.А. Полякова |
| (подпись, дата) |
| Прохождение нормоконтроля: |  | заведующий лабораториями  Д.А. Карлов |
| (подпись, дата) |
| Прохождение проверки на объем заимствований: |  | инженер  Е.В. Ерискина |
|  | (подпись, дата) |  |

Пермь — 2022Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

Пермский национальный исследовательский политехнический университет

(ПНИПУ)

Факультет: Электротехнический (ЭТФ)

Направление: 09.03.04 – Программная инженерия (ПИ)

Профиль: Разработка программно-информационных систем (РИС)

Кафедра информационных технологий и автоматизированных систем (ИТАС)

УТВЕРЖДАЮ

Зав. кафедрой ИТАС: д-р экон. наук, проф.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р.А. Файзрахманов

«\_\_\_\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

**ЗАДАНИЕ**

на выполнение выпускной квалификационной работы бакалавра

Фамилия, имя, отчество: Савин Сергей Алексеевич

Группа: РИС-18-1б

Начало выполнения работы: 01.09.2021

Контрольные сроки просмотра работы кафедрой:

1. 04.04.2022;
2. 25.04.2022;
3. 24.05.2022.

Защита работы на заседании ГЭК: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* + - 1. Наименование темы: «Разработка программного обеспечения задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM»
      2. Исходные данные к работе: техническое задание на разработку программного обеспечения задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM
      3. Содержание пояснительной записки:

1. Исследовательский раздел:

Концепция управления предприятием по целям, сбалансированная система показателей, ключевые показатели эффективности, описание бизнес-требований и функциональных требований, описание функциональных возможностей Websoft HCM, исследование устройства Websoft HCM, выводы по разделу;

1. Конструкторский раздел:

Выбор средств разработки, проектирование бизнес-процесса, проектирование пользовательского интерфейса, проектирование базы данных, проектирование системы разграничения доступа к информации, проектирование алгоритмов, выводы по разделу;

1. Технологический раздел:

Настройка окружения разработки, реализация пользовательских интерфейсов, реализация алгоритмов, выводы по разделу;

1. Раздел по защите информации:

Определение класса защищенности информационной системы и уровня значимости информации, примененные меры обеспечения информационной безопасности;

1. Экономический раздел:

Определение класса защищенности программной системы и уровня значимости информации, примененные меры обеспечения информационной безопасности;

* + - 1. Дополнительные указания: дополнительных указаний нет
      2. Основной список источников:

1. Афанасьева Л. А. Методологические аспекты управления мотивацией и стимулированием трудовой деятельности компании // Основы экономики, управления и права. – 2013. – № 5 (11) [Электронный ресурс]. – URL: http://cyberleninka.ru/article/n/metodologicheskie-aspekty-upravleniya-motivatsiey-i-stimulirovaniem-trudovoy-deyatelnosti-kompanii.
2. Мотивация персонала: учеб. пособие (практикум) / Ю. Г. Одегов, Г. Г. Руденко, С. Н. Апенько, А. И. Мерко. – М. : Альфа-Пресс, 2010. – 633 с.
3. Angular [Электронный ресурс] URL: https://angular.io/
4. Клиентский портал Websoft [Электронный ресурс] URL: https://news.websoft.ru/\_wt/default

|  |  |  |
| --- | --- | --- |
| Руководитель ВКР: |  | канд. техн. наук, доцент каф. ИТАС  О.А. Полякова |
| (подпись, дата) |
| Консультант по предметной области: |  | руководитель проектной группы А. А. Бартошевич. |
| (подпись, дата) |
| Задание получил: |  | студент  гр. РИС-18-1б  Савин Сергей Алексеевич |
| (подпись, дата) |

КАЛЕНДАРНЫЙ ГРАФИК ВЫПОЛНЕНИЯ

ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапа | Объем, % | Начало | Конец |
| 1. Сбор и анализ исходных данных, постановка задачи | 15 | 01.09.21 | 01.10.21 |
| 1. Изучение теоретического материала по предметной области; анализ и выбор методов и средств решения задачи | 20 | 02.10.21 | 01.10.21 |
| 1. Разработка теоретической части, методики решения; выбор и разработка средств решения задачи | 20 | 02.10.21 | 10.03.22 |
| 1. Тестирование разработки, постановка экспериментов, формулировка выводов | 20 | 11.04.22 | 04.05.22 |
| 1. Оформление пояснительной записки | 25 | 05.05.22 | 31.05.22 |
| 1. Представление работы на проверку и отзыв руководителя квалификационной работы | - |  |  |
| 1. Предварительная защита работы | - |  |  |
| 1. Прохождение нормоконтроля | - |  |  |
| 1. Представление работы на кафедру | - |  |  |
| 1. Защита на заседании ГЭК | - |  |  |

|  |  |  |
| --- | --- | --- |
| Руководитель ВКР: |  | канд. техн. наук,  доцент каф. ИТАС  О.А. Полякова |
| (подпись, дата) |
| Задание получил: |  | Савин Сергей Алексеевич |
| (подпись, дата) |

РЕФЕРАТ

Выпускная квалификационная работа бакалавра состоит из 5 разделов, 109 стр., 32 рис., 8 источн., 1 прил.

ОЦЕНКА КЛЮЧЕВЫХ ПОКАЗАТЕЛЕЙ ПРОИЗВОДИТЕЛЬНОСТИ, ЦЕЛЬ, КАРТА ЦЕЛЕЙ, ПЕРИОД ЦЕЛЕПОЛАГАНИЯ, ВЕБ-ПРИЛОЖЕНИЕ, ПЛАТФОРМА Websoft HCM, Angular

Объектом разработки является программное обеспечение задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM.

В результате проделанной работы выполнены следующие задачи: изучены и проанализированы концепция управления предприятием по целям, сбалансированная система показателей и ключевые показатели эффективности, разработаны бизнес-требования и функциональные требования к программному обеспечению для оценки и анализа ключевых показателей производительности работы сотрудников предприятия, исследовано устройство программной платформы Websoft HCM и ее функциональные возможности, спроектированы бизнес-процесс оценки и анализа ключевых показателей производительности работы сотрудников, схема базы данных и система разграничения доступа к информации, выбраны средства реализации программного обеспечения, спроектированы алгоритмы работы программного обеспечения, а также пользовательские интерфейсы для работы с ним, разработана система разграничения доступа к информации, разработаны алгоритмы работы программного обеспечения, Разработаны пользовательские интерфейсы, Определен класс защищенности информационной системы и уровень значимости информации, описаны применяемые меры обеспечения информационной безопасности.

Таким образом, результатом выполненной работы является спроектированное и разработанное программное обеспечение задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM

СОДЕРЖАНИЕ

[ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 8](#_Toc105954438)

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 9](#_Toc105954439)

[ВВЕДЕНИЕ 10](#_Toc105954440)

[1 Исследовательский раздел 12](#_Toc105954441)

[1.1 Концепция управления компанией по целям 12](#_Toc105954442)

[1.2 Сбалансированная система показателей 14](#_Toc105954443)

[1.3 Ключевые показатели эффективности 17](#_Toc105954444)

[1.4 Описание бизнес-требований и функциональных требований к разрабатываемому программному обеспечению задачи автоматизации процесса оценки и анализа ключевых показателей производительности работы сотрудников компании 19](#_Toc105954445)

[1.5 Описание функциональных возможностей Websoft HCM 20](#_Toc105954446)

[1.6 Исследование устройства Websoft HCM 23](#_Toc105954447)

[1.7 Вывод по разделу 25](#_Toc105954448)

[2 Конструкторский раздел 26](#_Toc105954449)

[2.1 Выбор средств разработки 26](#_Toc105954450)

[2.2 Проектирование бизнес-процесса 27](#_Toc105954451)

[2.3 Проектирование пользовательского интерфейса 28](#_Toc105954452)

[2.4 Проектирование базы данных 32](#_Toc105954453)

[2.5 Проектирование системы разграничения доступа к информации 38](#_Toc105954454)

[2.6 Проектирование алгоритмов 39](#_Toc105954455)

[2.6.1 Проектирование алгоритма создания и редактирования периода целеполагания 39](#_Toc105954456)

[2.6.2 Проектирование алгоритма создания карты целей 40](#_Toc105954457)

[2.6.3 Проектирование алгоритма создания и редактирования цели 41](#_Toc105954458)

[2.6.4 Проектирование алгоритмов переключения языка 44](#_Toc105954459)

[2.6.5 Проектирование алгоритма построения отчетной документации 46](#_Toc105954460)

[2.7 Выводы по разделу 47](#_Toc105954461)

[3 Технологический раздел 48](#_Toc105954462)

[3.1 Настройка окружения разработки 48](#_Toc105954463)

[3.1.1 Установка зависимостей 48](#_Toc105954464)

[3.1.2 Настройка форматирования кода с помощью Prettier 50](#_Toc105954465)

[3.2 Реализация пользовательских интерфейсов 51](#_Toc105954466)

[3.2.1 Реализация интерфейса для работы с периодами целеполагания 51](#_Toc105954467)

[3.2.2 Реализация интерфейса управления картами целей 57](#_Toc105954468)

[3.2.3 Реализация интерфейса управления картой цели сотрудника 66](#_Toc105954469)

[3.2.4 Реализация интерфейса построения отчетной документации 79](#_Toc105954470)

[3.3 Реализация алгоритмов 87](#_Toc105954471)

[3.3.1 Реализация алгоритма создания и редактирования периода целеполагания 88](#_Toc105954472)

[3.3.2 Реализация алгоритма создания карты целей 91](#_Toc105954473)

[3.3.3 Реализация алгоритма создания и редактирования цели 94](#_Toc105954474)

[3.3.4 Реализация алгоритмов переключения языка 98](#_Toc105954475)

[3.3.5 Реализация алгоритма построения отчетной документации 100](#_Toc105954476)

[3.4 Выводы по разделу 101](#_Toc105954477)

[4 Обеспечение информационной безопасности 102](#_Toc105954478)

[4.1 Определение класса защищенности информационной системы и уровня значимости информации 102](#_Toc105954479)

[4.2 Примененные меры обеспечения информационной безопасности 104](#_Toc105954480)

[5 Технико-экономическое обоснование 105](#_Toc105954481)

[5.1 Расчет затрат на разработку 105](#_Toc105954482)

[5.2 Расчет экономического эффекта 105](#_Toc105954483)

[ЗАКЛЮЧЕНИЕ 107](#_Toc105954484)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 109](#_Toc105954485)

[ПРИЛОЖЕНИЕ А 110](#_Toc105954486)

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящей работе применяют следующие термины с соответствующими определениями:

Angular – Фреймворк для создания одностраничных веб-приложений.

Angular-компонент – обособленная часть функционала со своей логикой, HTML-шаблоном и CSS-стилями.

Framework (рус. фреймворк) – программная платформа, которая упрощает создание и поддержку технически сложных или нагруженных проектов.

HTML-страница – документ или информационный интернет ресурс, доступ к которому осуществляется с помощью веб-браузера.

Websoft HCM – программная платформа для автоматизации бизнес-процессов предприятия.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящем отчете о выпускной квалификационной работе применяют

следующие сокращения и обозначения:

BSC – Balanced Scoreboard, сбалансированные показатели

CSS – Cascading Style Sheets, каскадные таблицы стилей

HTTP – Hypertext transfer protocol, протокол передачи гипертекста

HTML – HyperText Markup Language, язык гипертекстовой разметки

KPI – Key performance indicators, ключевые показатели эффективности

MBO – Management by objectives, управление по целям

UML – Unified Modeling Language, унифицированный язык моделирования

XML – eXtensible Markup Language, расширяемый язык разметки

ВВЕДЕНИЕ

Современным подходом к управлению бизнес-процессами компании является система параметров, которые помогают оценить степень достижения стратегических целей компании как общих, так и личных достижений сотрудников, ход выполнения проектных работ. (Key Performance Indicators – KPI).

KPI основаны на сбалансированной системе показателей(откуда), которая позволяет менеджерам компаний отследить степень исполнения сотрудниками поставленных задач.

Сбалансированная система показателей является методом стратегического управления, которая реализует комплексный подход для достижения глобальных целей компании. Реализация процесса достижения поставленных целей основана на декомпозиции задач верхнего уровня на подзадачи, решение которых в совокупности обеспечивает выполнение заданных целевых показателей таким образом, что достижение глобальной цели компании происходит автоматически через решение задач нижних уровней.

KPI контролирует активность сотрудников, отделов, подразделений и компании.

Компании необходимо наблюдать за своими бизнес-процессами в режиме реального времени, для этого разрабатываются программные экосистемы, которые позволяют хранить, анализировать и управлять большими объемами данных. Для этих целей компания-заказчик использует программную платформу Websoft HCM.

**Объектом** исследования является оценка и анализ ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM.

**Предметом** исследования – программное обеспечение для автоматизации оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM.

**Целью** данной работы является проектирование и разработка программного обеспечения задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM.

Для достижения поставленной цели определены **задачи**:

* изучить и проанализировать концепцию управления предприятием по целям, сбалансированную систему показателей и ключевые показатели эффективности;
* разработать бизнес-требования и функциональные требования к программному обеспечению для оценки и анализа ключевых показателей производительности работы сотрудников предприятия;
* исследовать устройство программной платформы websoft hcm и ее функциональные возможности;
* спроектировать бизнес-процесс оценки и анализа ключевых показателей производительности работы сотрудников, схему базы данных и систему разграничения доступа к информации;
* выбрать средства реализации программного обеспечения и настроить окружение разработки;
* спроектировать алгоритмы работы программного обеспечения, а также пользовательские интерфейсы для работы с ним;
* разработать систему разграничения доступа к информации;
* разработать алгоритмы работы программного обеспечения;
* разработать пользовательские интерфейсы;
* определить класс защищенности информационной системы и уровень значимости информации, описать применяемые меры обеспечения информационной безопасности.

# Исследовательский раздел

## Концепция управления компанией по целям

Для эффективного управления компанией необходимо точно сформировать цели, а также разработать систему достижения этих целей, определять, и измерять показатели их выполнения.

Финансовый успех компании в полной мере зависит от индивидуальных усилий и ответственности всех его работников и будет выше, если интересы компании и сотрудников в целом совпадают[1].

Основоположником концепции управления по целям (MBO – Management by objectives) является Питер Дуркер: «Менеджмент компании должен контролировать ее экономические ресурсы и воздействовать на них осознанно и целенаправленно.»

Стратегические цели компании – это система ориентиров компании в долгосрочном периоде, суть которой состоит в работе менеджмента компании по постановке и донесении целей до сотрудников, обеспечении их необходимыми ресурсами, а также распределении ролей и ответственности за достижение поставленных целей[2].

Основные принципы MBO:

* цели ставятся на уровне организации и для каждого ее сотрудника;
* при разработке целей обеспечивается связь со стратегией компании;
* цели должны быть понятны и достижимы сотрудниками компании;
* цели разрабатываются руководством компании и подчиненными; при таком подходе на всех уровнях понятно, что делать и как делать;
* оценка проделанной работы производится руководством оперативно;
* подчиненные получают постоянную обратную связь от руководства;
* цели отвечают подходу smart.

SMART – современный подход к постановке целей, которые должны отвечать определенным принципам.

Подробное описание SMART-целей представлено в таблице 1.1.

Таблица 1.1 – Описание SMART-целей

|  |  |
| --- | --- |
| Критерий SMART |  |
| Specific (Конкретный) | Цель:   * Четко сформулирована; * Направлена на конкретный результат; * Имеет ограничения по реализации. |
| Measurable (Измеримый) | Измерение степени достижения цели производится при помощи конкретных критериев |
| Achievable (Достижимый) | Цель реалистична. Учитываются квалификация, личные качества персонала, ресурсы компании |
| Relevant  (Значимый) | Вклад, который обеспечит достижение цели в решении глобальных задач компании |
| Time bound (Ограниченный по времени) | Выполнение цели ограничено временными рамками |

## Сбалансированная система показателей

Концепция MBO направлена на повышение финансовых показателей компании. Снижение расходов на маркетинг, обучение персонала, обслуживание клиентов в целях улучшения краткосрочных финансовых показателей негативно сказывается на благополучии компании. Именно поэтому данная концепция модифицирована Дэвидом Нортоном и Робертом Капланом, которые интегрировали в MBO сбалансированную систему показателей (BSC – Balanced Scoreboard).

При использовании данной системы стратегические цели компании сгруппированы по четырем направлениям:

* финансы;
* клиенты;
* развитие и обучение персонала;
* внутренние бизнес-процессы.

Для каждого из направлений подбирается набор качественных и количественных показателей – метрик для оценки степени достижения поставленных целей.

Качественные показатели измеряются с помощью экспертных оценок наблюдения за процессом и результатами работы.

Количественные показатели выражаются в виде некоторого действительного числа, имеющего определенный физический или экономический смысл.

Примеры количественных и качественных показателей представлены в таблице 1.2.

Таблица 1.2 – Соответствие направлений и показателей

|  |  |  |
| --- | --- | --- |
| Направление | Вид показателя | Показатели |
| Финансы | Количественные | Чистая прибыль, издержки, дебиторская задолженность, объем продаж, денежный поток |
| Клиенты | Количественные | Доля своевременных поставок клиенту, объем продаж новым клиентам |
| Качественные | Индекс удовлетворенности клиентов |
| Внутренние бизнес-процессы | Количественные | Время обслуживания, число отказов системы |
| Качественные | Уровень трудовой и исполнительской дисциплины, соблюдение стандартов и регламента |
| Развитие и обучение персонала | Количественные | Текучесть кадров, заболеваемость |
| Качественные | Индекс удовлетворенности сотрудников, квалификация сотрудников |

Сбалансированная система показателей BSC:

* повышает степень информированности руководителей об эффективности деятельности компании;
* информирует персонал о намеченных стратегических целях и приоритетных направлениях развития;
* информирует сотрудников компании о роли каждого в достижении поставленных целей.

Охват основных сфер деятельности компании, финансовых показателей, а также показателей работы с клиентами и персоналом, внутренних бизнес-процессов являются преимуществами, которые обеспечиваются внедрением системы BSC.

Внедрение такой масштабной системы в компании требует больших временных и финансовых ресурсов.

## Ключевые показатели эффективности

Система ключевых показателей эффективности (KPI) измеряет результативность и эффективность проведенных работ, помогает руководителям правильно организовывать производственный процесс: анализировать результаты деятельности сотрудников, корректировать их действия, распределять вознаграждение за работу.

KPI активно используются для оценки качества обучения и квалификации персонала. Это обусловлено тем, что технологии стремительно развиваются, а следовательно, меняются требования к профессиональной подготовке сотрудников. Без детализированной системы ключевых показателей невозможно оценить эффективность проводимых мероприятий по обучению и повышению квалификации сотрудников.

Система KPI соответствует стратегическим целям компании и учитывает специфику ее деятельности.

Примеры ключевых показателей эффективности представлен представлены в таблице 1.3.

Таблица 1.3 – Примеры ключевых показателей эффективности

|  |  |
| --- | --- |
| KPI | Пояснение |
| Индекс вовлеченности персонала | Высокий уровень данного показателя положительно влияет на продуктивность и качество работы с клиентами, снижает текучесть кадров.  где n – количество сотрудников, ki – баллы за опрос сотрудника, m – максимальное количество баллов за опрос |

Продолжение таблицы 1.3

|  |  |
| --- | --- |
| KPI | Смысловая нагрузка |
| Качество найма | Данный KPI отражает долю новых сотрудников, которые получили положительные отзывы от руководителей во время оценки эффективности.  Качество найма отражает насколько эффективна работа HR-отдела компании. Стабильно высокий уровень данного показателя позволяет организации быстрее достигать своих стратегических целей.  где k – качество найма, n – количество новых сотрудников, получившие положительный отзыв от руководителей, m – количество новых сотрудников |
| Коэффициент нежелательной текучести | Показатель отражает долю сотрудников, покинувших компанию, чьи рабочие показатели были высоки.  где k – коэффициент нежелательной текучести, n – ушедшие сотрудники, чьи показатели эффективности высоки, m – количество ушедших сотрудников |
| Обеспечение необходимого качества обучения | Оценка знаний обученных сотрудников, а также их удовлетворенности после обучения.  где k – коэффициент обеспечения необходимо качества обучения, n – итоговый балл сотрудника за тест, m – итоговый балл сотрудника за тест после обучения |

Продолжение таблицы 1.3

|  |  |
| --- | --- |
| KPI | Смысловая нагрузка |
| Доля стоимости рабочей силы | KPI применяется для анализа расходов и автоматизации процессов компании.  где k – доля стоимости рабочей силы, n – расходы на сотрудников, m – общие расходы компании |

## Описание бизнес-требований и функциональных требований к разрабатываемому программному обеспечению задачи автоматизации процесса оценки и анализа ключевых показателей производительности работы сотрудников компании

Цель разрабатываемой системы – упрощение планирования целей компании, снижение нагрузки на менеджмент, повышение вовлеченности персонала (через денежное стимулирование) в деятельность компании, повышение качества отчетной документации по производительности работы сотрудников.

Исходя из вышеперечисленных целей разработаны:

1. Бизнес-требования, а именно:

* иерархия целей;
* методы контроля доступа к информационным ресурсам системы;
* система автоматического сбора и анализа данных о производительности сотрудников и подразделений компании.

1. Функциональные требования:

* встраивание ПО в программную платформу Websoft HCM;
* автоматическое разграничение функционала по ролям доступа;
* система автоматически контролирует периоды целедостижения;
* автоматическое формирование карт целей для каждого сотрудника;
* разработка количественных и качественных показателей для измерения степени достижения целей;
* автоматическое выстраивание иерархии целей;
* обеспечение связи между сотрудником и руководством: через системы рассылки уведомлений и комментирования целей;
* разработка системы для личных целей сотрудников;
* обеспечение обратной связи между сотрудниками компании;
* разработка автоматического построения отчетной документации;
* поддержка мультиязычности.

## Описание функциональных возможностей Websoft HCM

Доступ к функциональным возможностям платформы обеспечивается двумя интерфейсами: порталом Websoft для организации единой точки доступа пользователей к различным ресурсам и сервисам системы и приложением Administrator Websoft HCM, которое представляет собой автоматизированное рабочее место администратора для настройки системы и внесения новых программных модулей[3].

Функциональные возможности панели администратора представлены в виде UML-диаграммы прецедентов на рисунке 1.1.

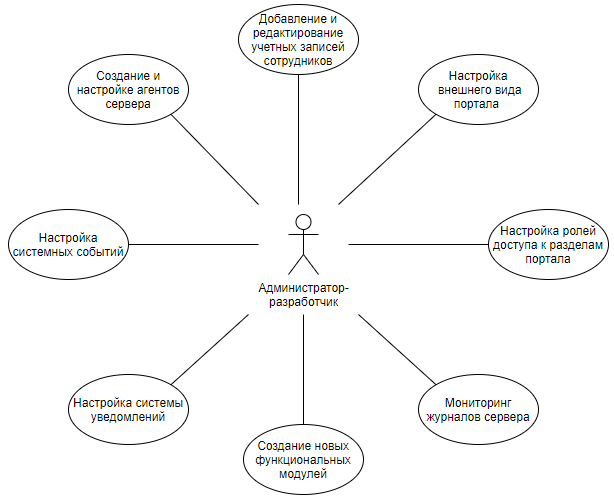


Рисунок 1.1 – Диаграмма прецедентов функциональных возможностей панели администратора

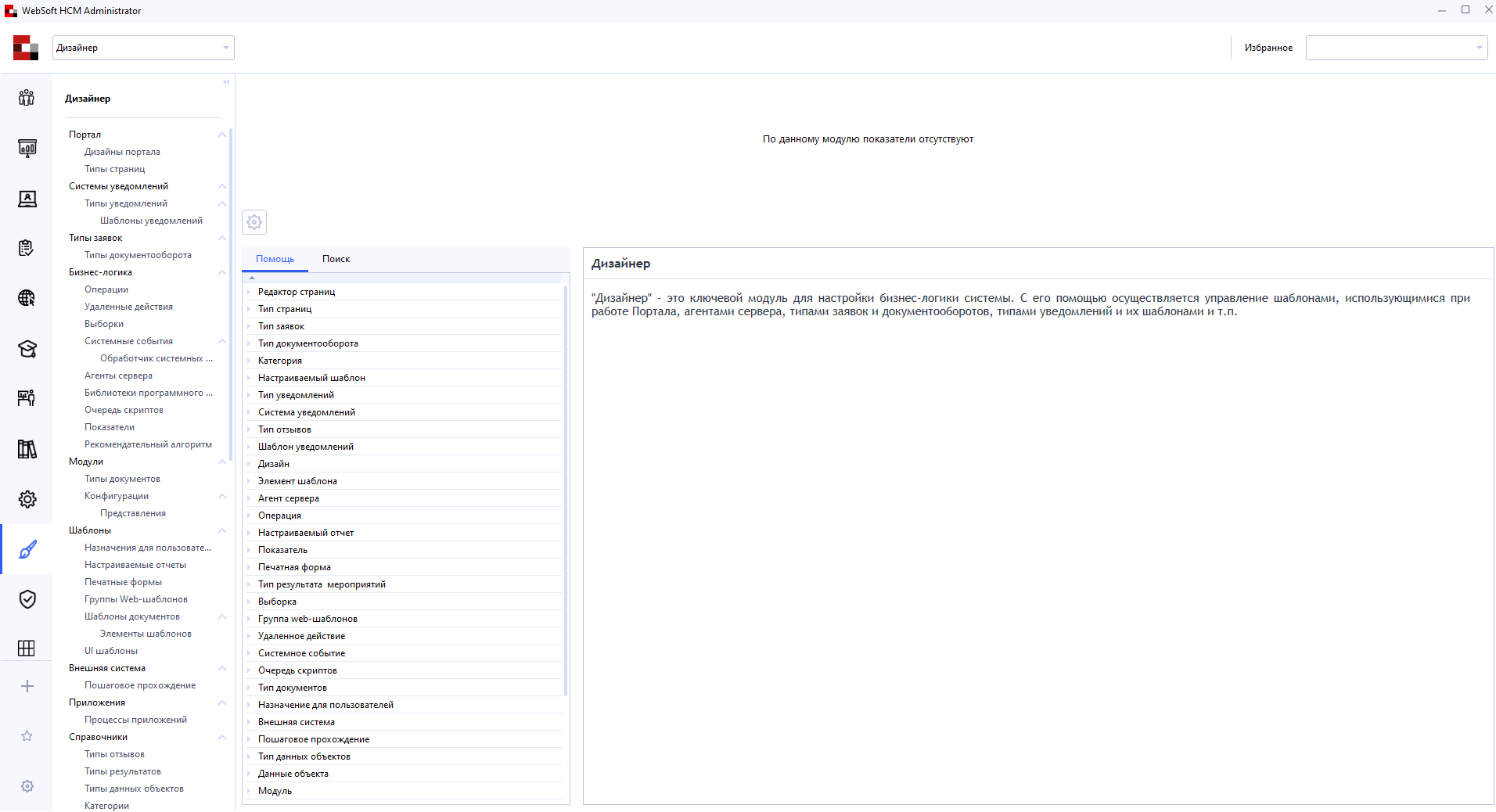
Панель администратора представлена на рисунке 1.2.

Рисунок 1.2 – Панель администратора

Доступ к функциональным возможностям портала предоставляется пользователю в соответствии с его ролью. Функциональны возможности портала отображены в виде UML-диаграммы прецедентов на рисунке 1.3.

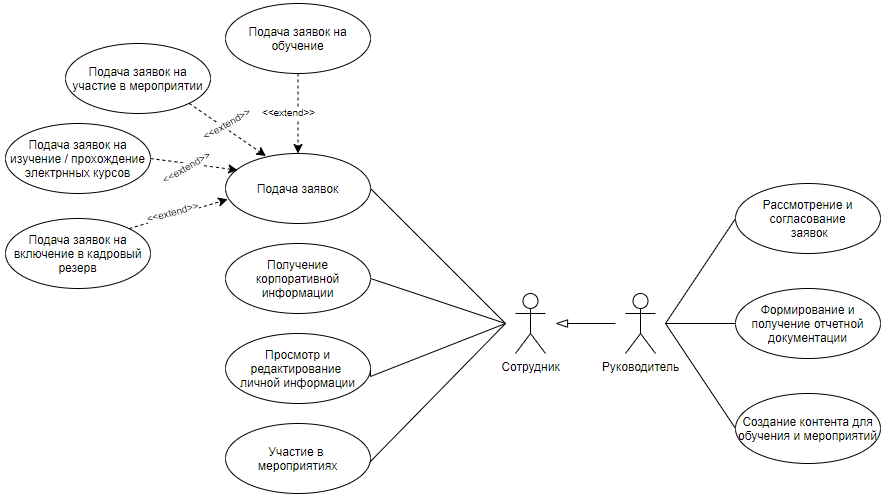


Рисунок 1.3 – Диаграмма прецедентов функциональных возможностей портала

Интерфейс информационного портала представлен на рисунке 1.4.

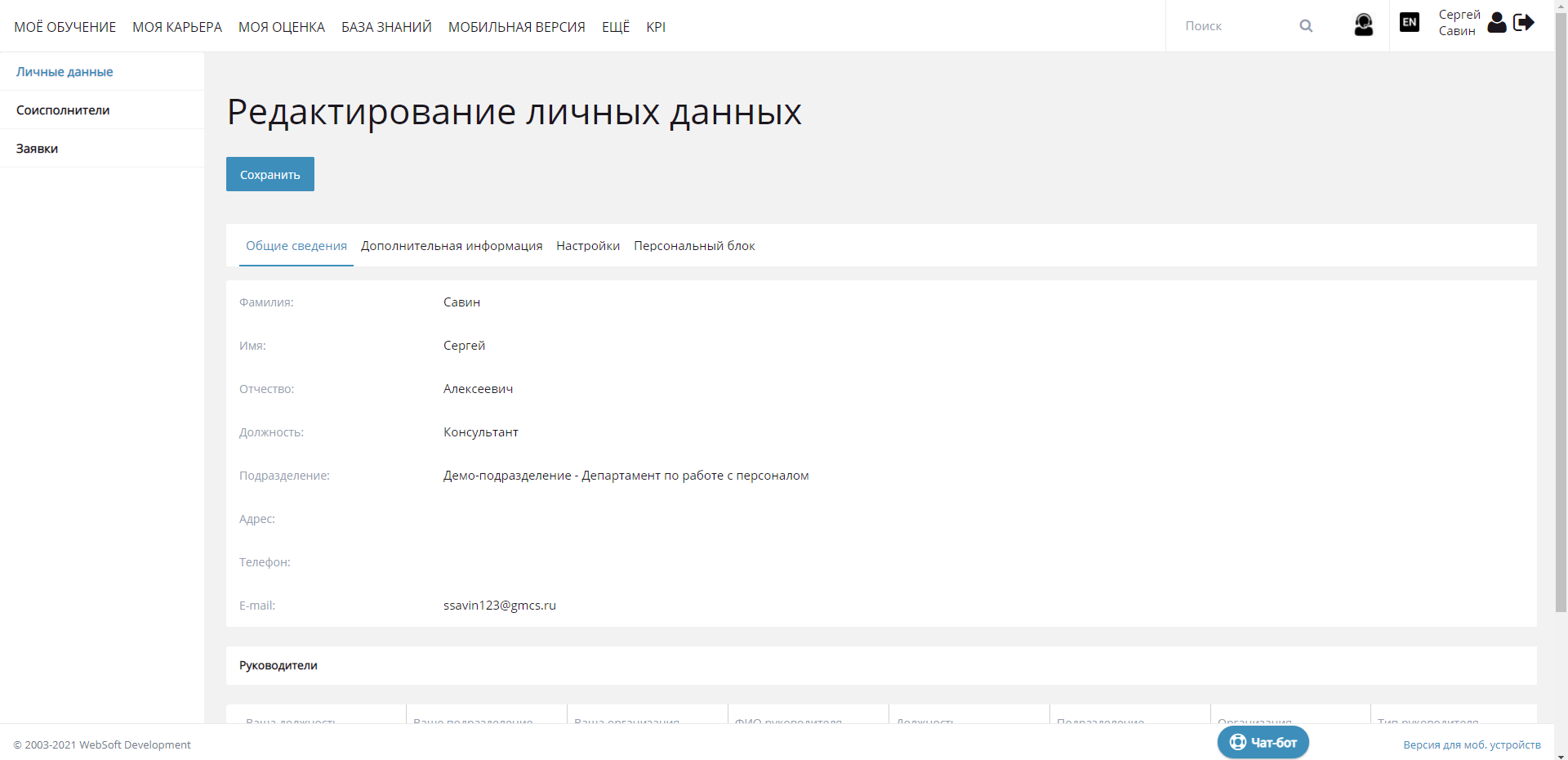


Рисунок 1.4 – Информационный портал

Преимуществами данной платформы являются: гибкая комплектация, интеграция с внешними IT-система, поддержка операционных систем Windows и Linux, единая база данных для всех бизнес-процессов, а также простота доработки, форум, wiki-база и постоянная техническая поддержка.

В то же время недостатками Websoft HCM являются: высокая стоимость платформы и дополнительных модулей для нее, сложный интерфейс и платаная техническая поддержка после второго года использования. Но данные факторы не влияют на решения заказчика по разработке данной системы.

## Исследование устройства Websoft HCM

Работа Websoft HCM устроена на собственной платформе SP-XML. Данная платформа включает в себя:

* средства описания структуры данных (на основе xml);
* встроенную СУБД;
* встроенный интерпретатор JavaScript;
* сервер, позволяющий строить клиент-серверные приложения;
* средства интеграции с другими приложениями и системами;
* средства для работы с сетевыми протоколами (HTTP, SMTP, POP3, IMAP).

Приложения, построенные на SP-XML, отвечают принципу открытости. Все структуры данных, код приложения, не входящий в ядро платформы, находятся в открытом виде в файлах, имеющих расширения xml, html, js, xaml, которые можно изменить через любой текстовый редактор.

Также приложения, построенные на данной платформе, отвечают принципу отделения данных от их представления в пользовательском интерфейсе.

SPXML предоставляет ряд функций для работы с различными типами данных, документами и элементами XML, файловой системой, сервером приложений.

Практически все взаимодействие с Websoft HCM осуществляется посредством протоколов HTTP и HTTPS.

Взаимодействие сервера система и СУБД происходит через программный интерфейс ODBC. Среди поддерживаемых СУБД можно выделить: SQL Server, Oracle, PostgreSQL.

Для отправки различных сообщений пользователем системы используется протокол SMTP. Почтовый сервер выступает в роли ретранслятора между сервером Websoft HCM и адресатом.

Архитектура платформы Websoft HCM представлена в виде контейнер-диаграммы нотации C4 на рисунке 1.5[4].

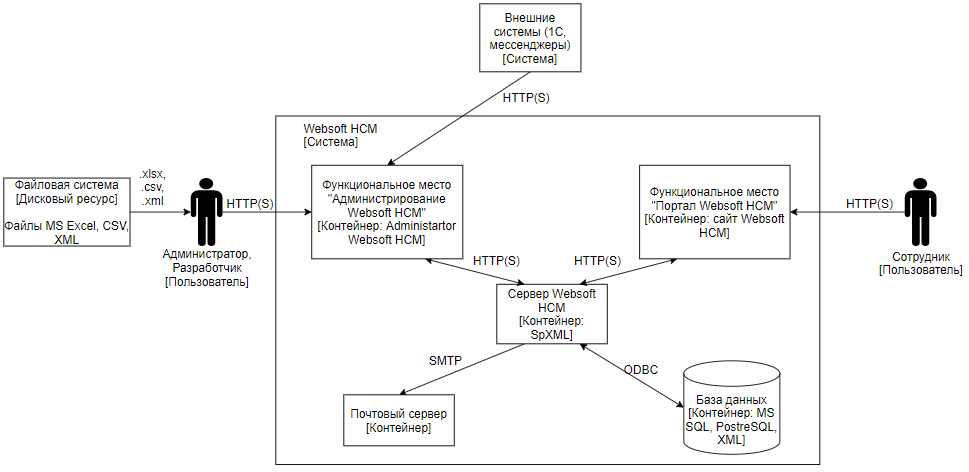


Рисунок 1.5 – Контейнер-диаграмма архитектуры платформы Websoft HCM

## Вывод по разделу

Таким образом, в данном разделе решены следующие задачи:

* выполнен анализ систем управления компанией по целям, сбалансированной системы показателей и системы ключевых показателей эффективности;
* сформированы функциональные требования и бизнес-требования к системе;
* исследованы устройство и функциональные возможности программной платформы Websoft HCM.

# Конструкторский раздел

## Выбор средств разработки

Для разработки серверной части приложения выбран язык программирования server-side JavaScript. Данное решение обусловлено особенностью платформы SP-XML.

В качестве системы управления базами данных выбран SQL Server.

Для реализации клиентской части приложения выбраны:

* язык HTML для создания разметки страницы;
* язык программирования TypeScript для программирования логики работы пользовательского интерфейса;
* препроцессор SCSS, предназначенный для упрощения создания CSS кода;
* фреймворк Angular, предназначенный для создания одностраничных клиентских приложений с использованием HTML и TypeScript[5].

## Проектирование бизнес-процесса

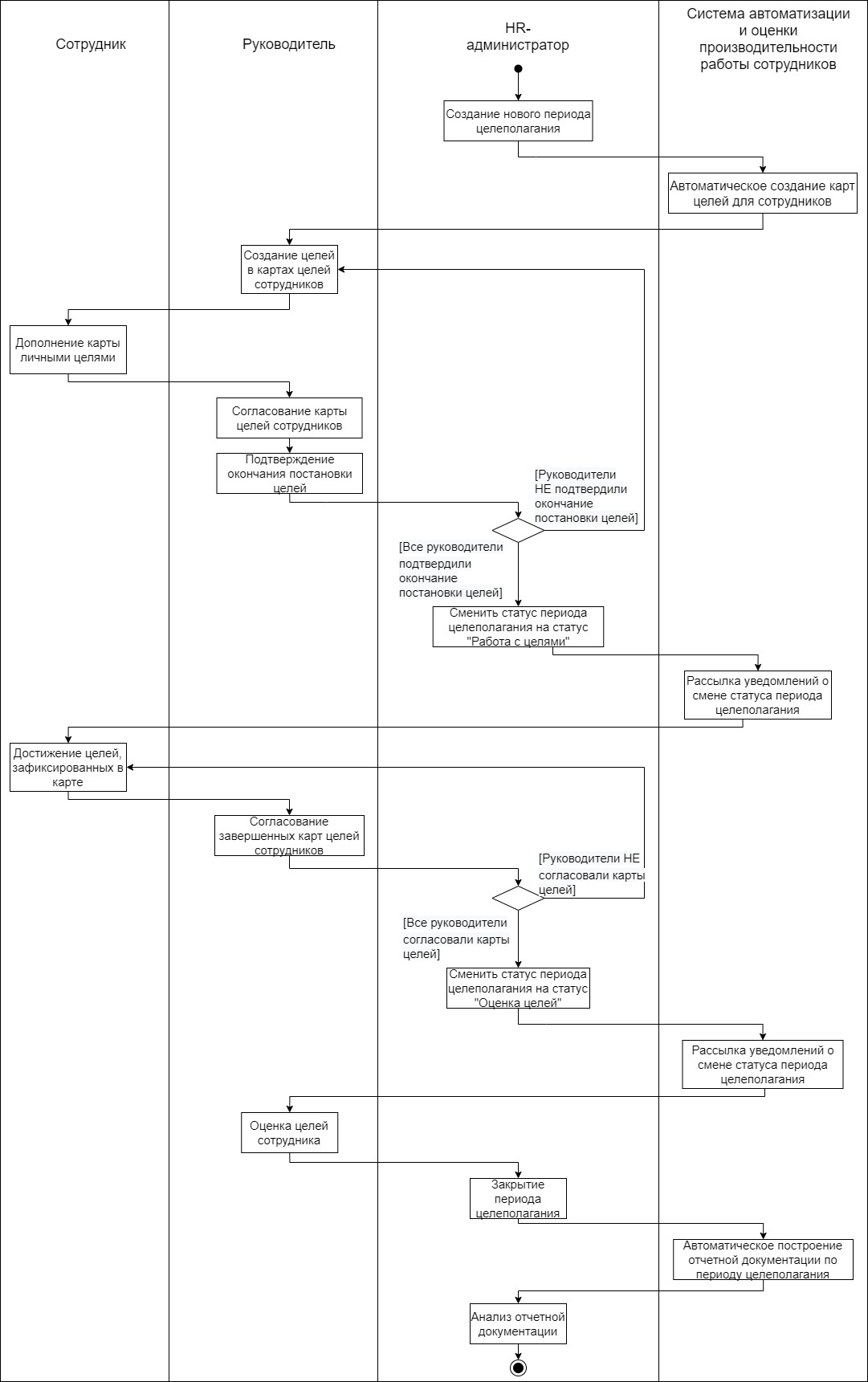
В рамках ВКР разработана и реализована модель бизнес-процесса, которая представлена в виде UML диаграммы деятельности на рисунке 2.1.

Рисунок 2.1 – Модель бизнес-процесса

При проектировании бизнес-процесса проанализирован опыт компаний, внедривших систему KPI.

## Проектирование пользовательского интерфейса

Для взаимодействия сотрудников компании с системой спроектирован пользовательский интерфейс, состоящий из 6 веб-окон. Доступ к веб-окном разграничен по ролям.

Для работы с периодами целеполагания разработано окно, представленное на рисунке 2.2.

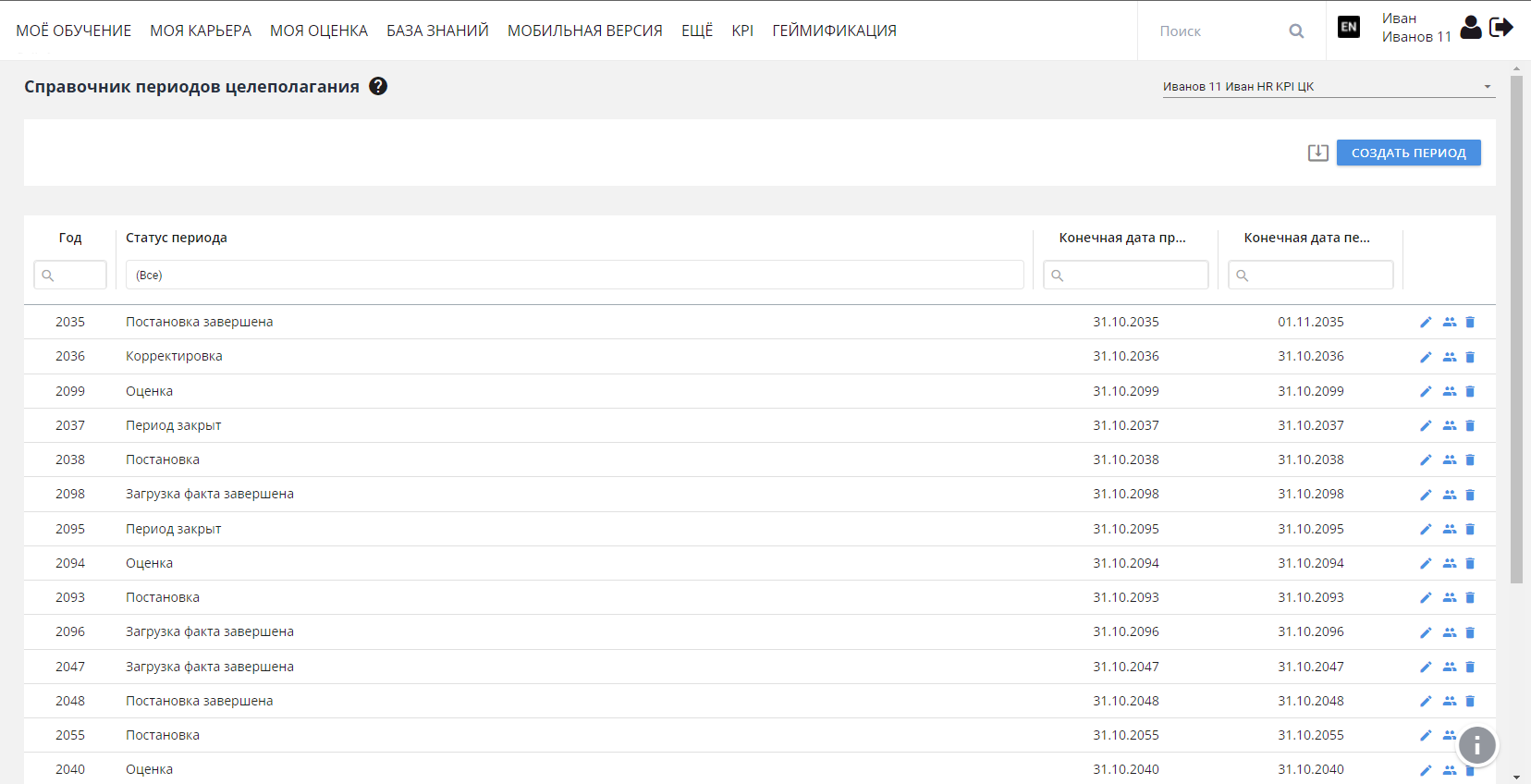


Рисунок 2.2 – Окно работы с периодами целеполагания

Главным элементом данного экрана является таблица, в которой отображены данные о периодах целеполагания: год, статус, конечные даты приема и перевода периода.

Конечная дата приема – это крайняя дата приема сотрудника. Если сотрудник принят после этой даты, то он не участвует в процессе целеполагания. Конечная дата перевода – это крайняя дата перевода сотрудника. Если сотрудник переведен после этой даты, то он не участвует в процессе целеполагания.

Также данная таблица содержит кнопки, для редактирования, удаления периода и просмотра информации о сотрудниках, которые не подтвердили окончание работ с целями. Окно имеет кнопку для создания нового периода целеполагания.

На рисунке 2.3 представлено окно для управления картами целей.

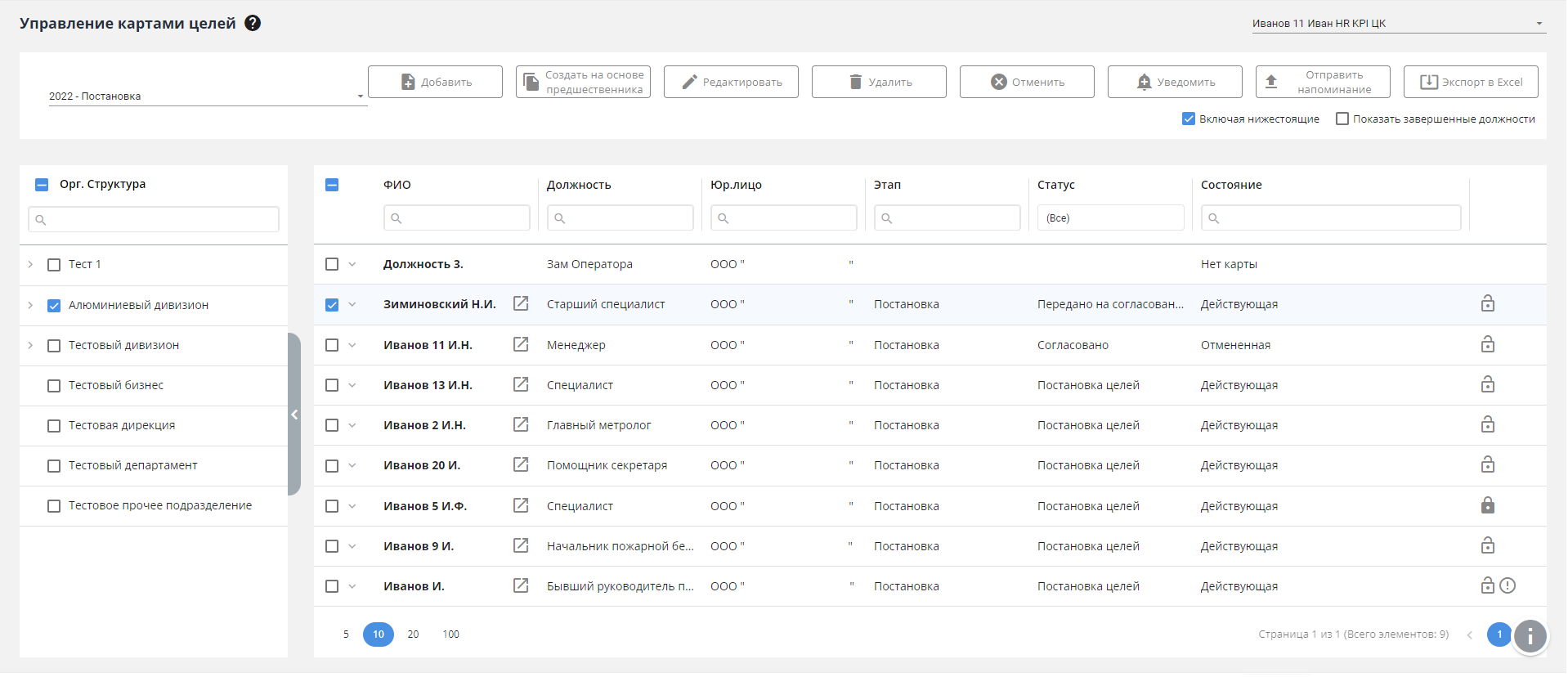


Рисунок 2.3 – Окно управления картами целей

Данное окно предназначено для управления картами цели: редактирование, удаление, создание. Также в данном окне можно отправить уведомление сотруднику нажав на кнопку «Уведомить». Форма отправки уведомления сотруднику представлена на рисунке 2.4.

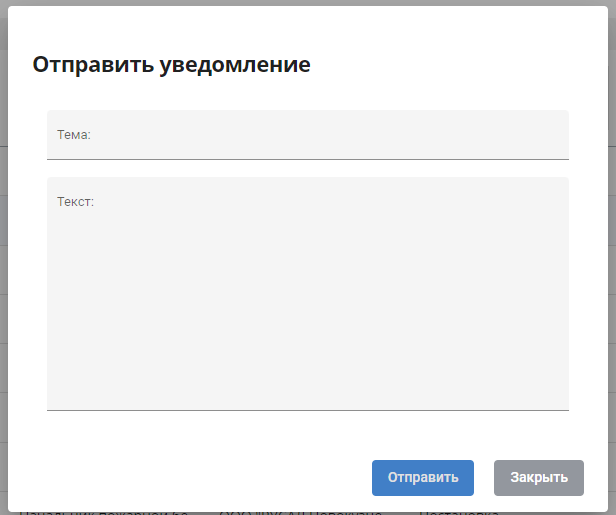


Рисунок 2.4 – Форма отправки уведомления

Из окна управления картами целей можно перейти в карту сотрудника, для того чтобы поставить цели или отредактировать их. Переход в карту целей сотрудника осуществляется посредством нажатия на иконку рядом с ФИО сотрудника.

Интерфейс окна управления картой цели сотрудника представлен на рисунке 2.5.

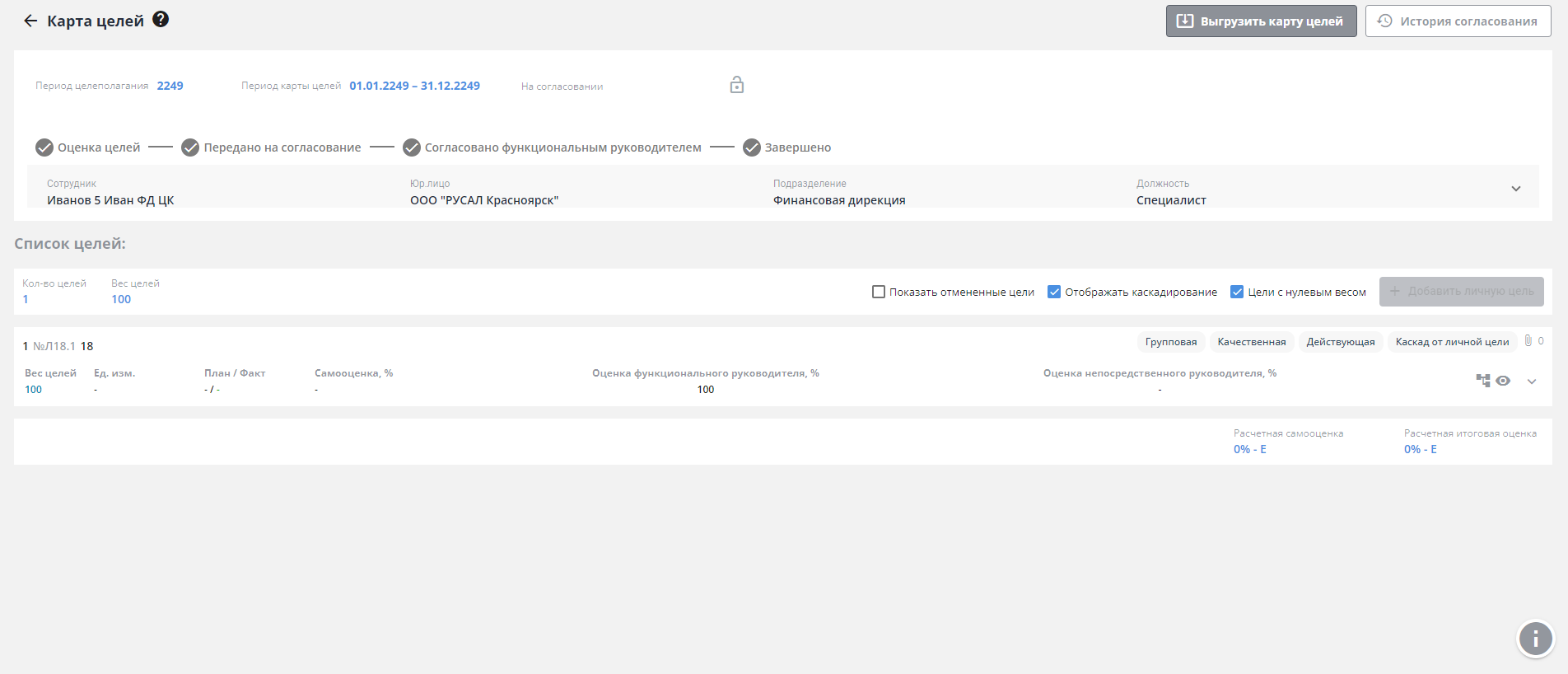


Рисунок 2.5 – Окно управления картой цели сотрудника

На данной странице можно посмотреть информацию о карте цели, историю согласования, цели, назначенные сотруднику, а также подробную информацию о них. Модальное окно просмотра истории согласования карты целей представлено на рисунке 2.6.

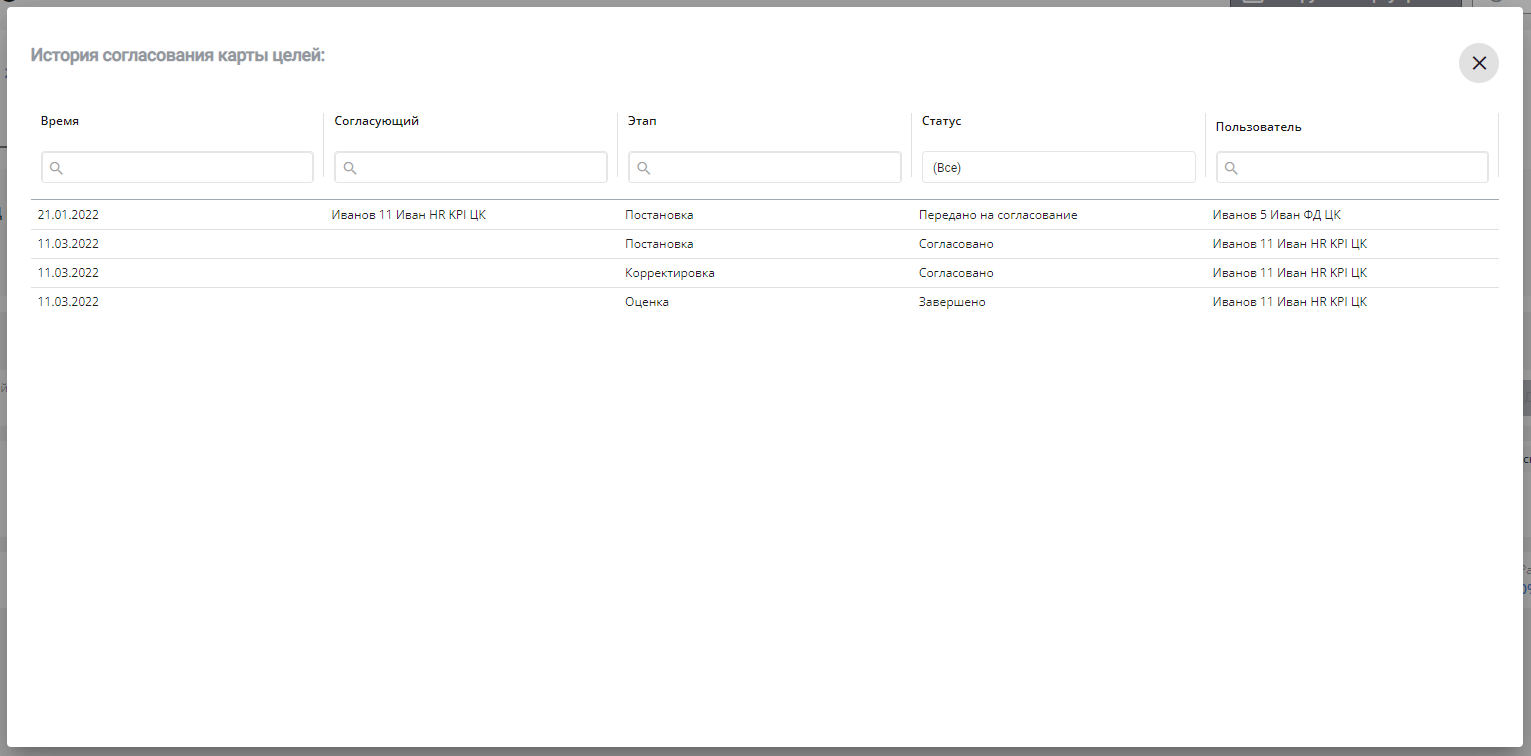


Рисунок 2.6 – Модальное окно просмотра истории согласования карты целей

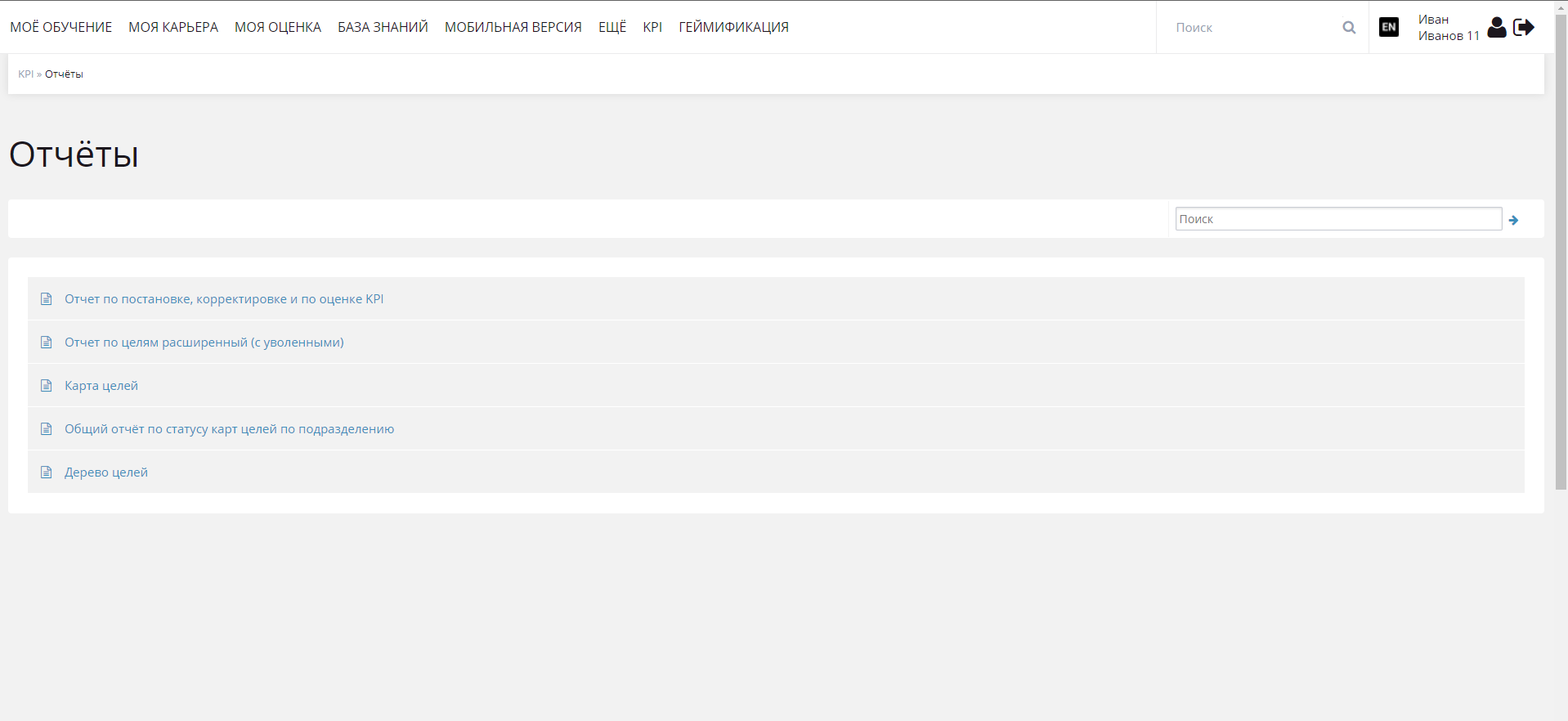
Страница, предоставляющая доступ к отчетам представлена на рисунке 2.7.

Рисунок 2.7 – Страница выбора отчета

На данной странице отображается список доступных отчетов пользователя.

Пример построенного отчета представлен на рисунке 2.8.

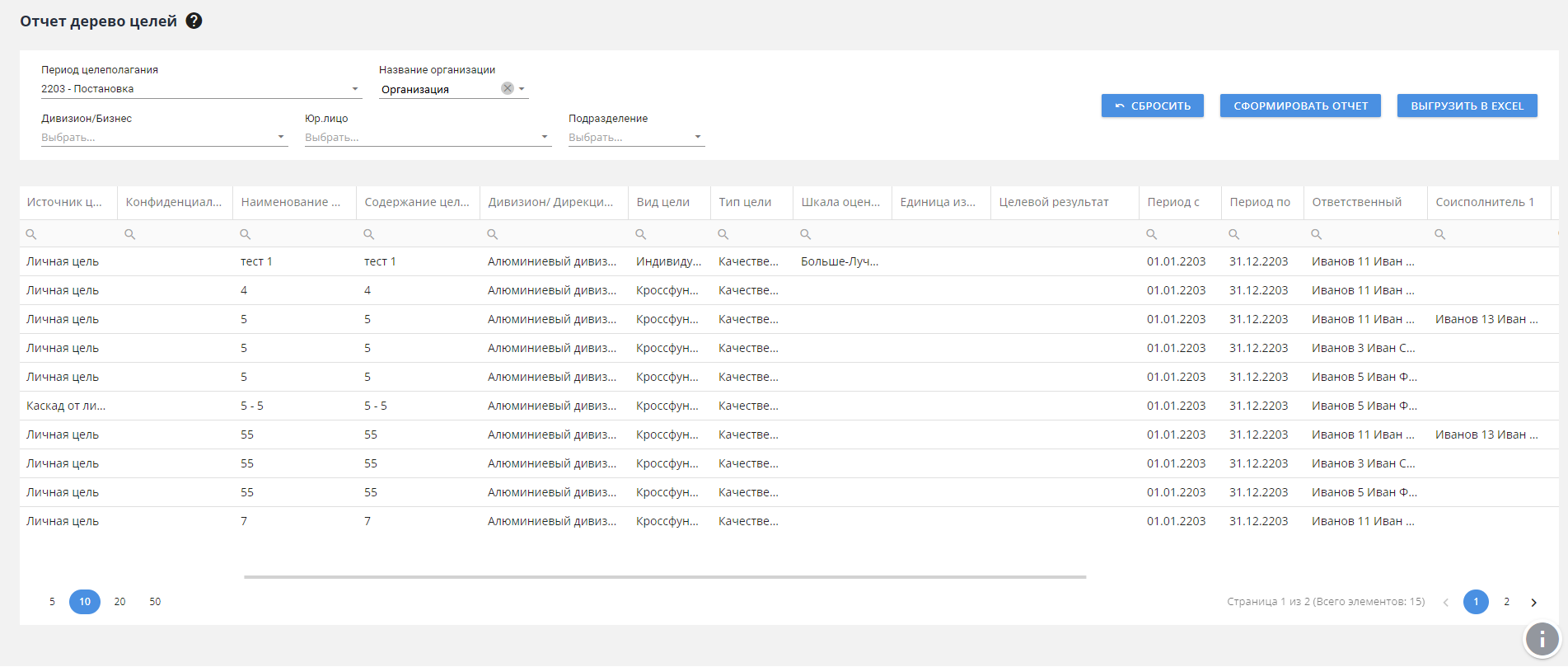


Рисунок 2.8 – Пример построенного отчета

На странице присутствуют фильтры для получения информации в разрезе организации, дивизиона, юридического лица, подразделения, сотрудника.

## Проектирование базы данных

Модель базы данных в виде ER‑диаграммы, выполненной в нотации Джеймса Мартина представлена на рисунке 2.9[6].

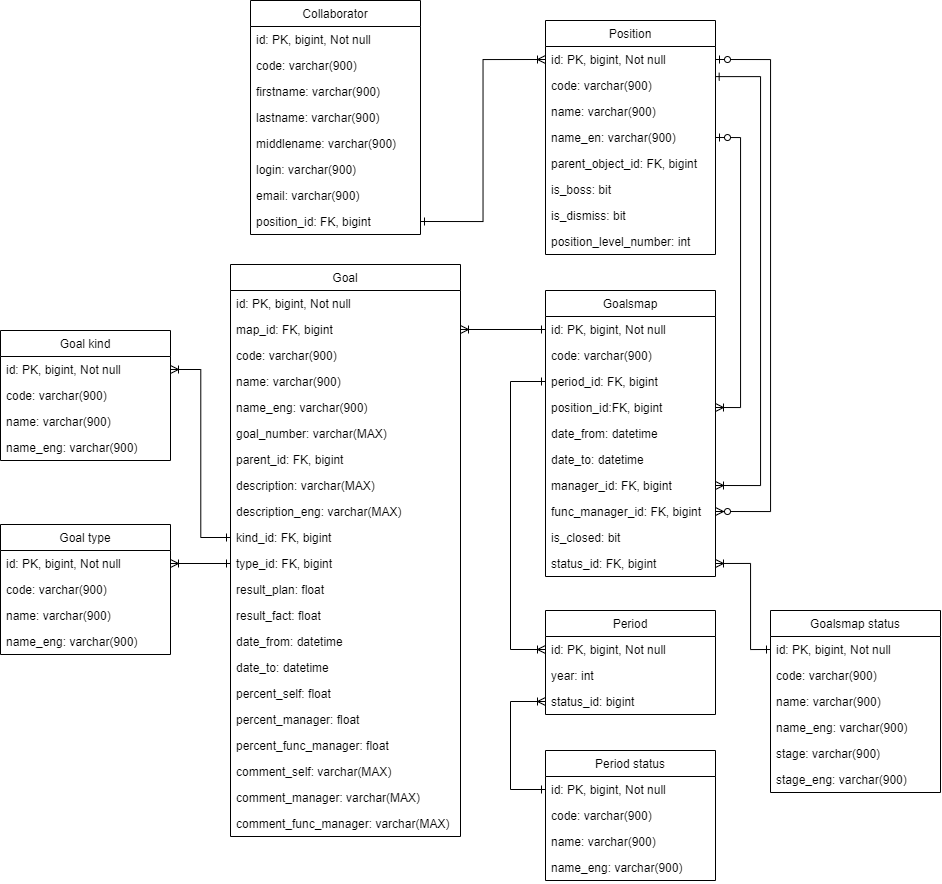


Рисунок 2.9 – Модель базы данных

Описание сущностей модели базы данных представлено в таблице 2.2.

|  |  |
| --- | --- |
| Сущность | Описание полей |
| Collaborator (Сотрудник) | id – Уникальный идентификатор сотрудника  firstname, lastname, middlename – Фамилия, имя, отчество сотрудника соответственно  login – Имя пользователя  email – Почта сотрудника  position\_id – Уникальный идентификатор должности, занимаемой сотрудником |
| Position (Должность) | id – Уникальный идентификатор должности  name, name\_en – Название должности на русском и английском языках соответственно  parent\_object\_id – Уникальный идентификатор подразделения, за которым закреплена должность  is\_boss – Флаг, указывающий является ли должность руководящей в подразделении  is\_dismiss – Флаг, указывающий является ли сотрудник занимающий должность уволенным  position\_level\_number – Число, отображающее уровень должности |

Таблица 2.2 – Описание сущностей БД

|  |  |
| --- | --- |
| Сущность | Описание полей |
| Goalsmap (Карта целей) | id – Уникальный идентификатор карты целей  period\_id – Уникальный идентификатор периода целеполагания, к которому относится карта  date\_from, date\_to – Даты действия карты  manager\_id – Уникальный идентификатор руководителя  manager\_id – Уникальный идентификатор руководителя |
| Goalsmap status (Статус карты целей) | id – Уникальный идентификатор статуса карты цели  name, name\_eng – Названия статусов на русском и английском языках соответственно  stage, stage\_eng – Названия этапа целеполагания на русском и английском языках соответственно |
| Period (Период целеполагания) | id – Уникальный идентификатор периода целеполагания  year – Год периода целеполагания  status\_id – Уникальный идентификатор статуса периода целеполагания |
| Period status (Статус периода целеполагания) | id – Уникальный идентификатор статуса периода целеполагания |

Продолжение таблицы 2.2

Продолжение таблицы 2.2

|  |  |
| --- | --- |
| Сущность | Описание полей |
|  | name, name\_eng – Названия статуса на русском и английском языках соответственно |
| Goal (Цель) | id – Уникальный идентификатор цели  map\_id – Уникальный идентификатор карты целей, к которой относится цель  name, name\_eng – Название цели на русском и английском языках соответственно  goal\_number – Номер цели  parent\_id – Уникальных идентификатор родительской цели  description, description\_eng – Описание цели на русском и английском языках соответственно  kind\_id – Уникальный идентификатор вида цели  type\_id – Уникальный идентификатор типа цели  result\_plan – Планируемый результат  result\_fact – Фактический результат  date\_from, date\_to – Даты действия цели  percent\_self, percent\_manager, percent\_func\_maanger – Самооценка, оценка непосредственного руководителя, оценка функционального руководителя по цели |

Продолжение таблицы 2.2

|  |  |
| --- | --- |
| Сущность | Описание полей |
|  | comment\_self, comment\_manager, comment\_func\_manager – Личный комментарий, комментарий непосредственного руководителя, комментарий функционального руководителя |
| Goal kind (Вид цели) | id – Уникальный идентификатор вида цели  name, name\_eng – Название на русском и английском языках соответственно  Возможные значения: количественная, качественная |
| Goal type (Тип цели) | id – Уникальный идентификатор типа цели  name, name\_eng – Название на русском и английском языках соответственно  Возможные значения: личная, групповая, кросс-функциональная |

## Проектирование системы разграничения доступа к информации

Для разграничения доступа к информационным и функциональным возможностям системы спроектированы роли доступа, представленные в таблице 2.3.

Таблица 2.3 – Роли доступа

|  |  |
| --- | --- |
| Роль | Описание роли |
| HR-администратор KPI БЕ | Пользователи, которые управляют процессами целеполагания в рамках своей бизнес единицы |
| HR-администратор KPI ЦК | Пользователи, которые управляют процессами целеполагания в рамках всей группы компаний.  В том числе ведут справочники для процесса целеполагания |
| Непосредственный руководитель | Непосредственный руководитель сотрудника |
| Сотрудник | Сотрудник, участвующий в целеполагании, ответственный за достижение целей, указанных в его карте целей |
| Сотрудник ФД БЕ | Сотрудник Финансовой дирекции.  Согласовывает приказные цели в рамках своей бизнес единицы. При необходимости, корректирует данные по целям |
| Сотрудник ФД ЦК | Сотрудник Финансовой дирекции.  Согласовывает приказные цели в рамках всей группы компаний. При необходимости, корректирует данные по целям |
| Функциональный руководитель | Сотрудник любого подразделения, который осуществляет функциональное руководство над сотрудником в соответствии с производственной необходимостью |
| Администратор | Администратор системы |

## Проектирование алгоритмов

Проектирование алгоритма создания и редактирования периода целеполагания

Алгоритм создания и редактирования периода целеполагания представлен в виде диаграммы деятельности на рисунке 2.10.

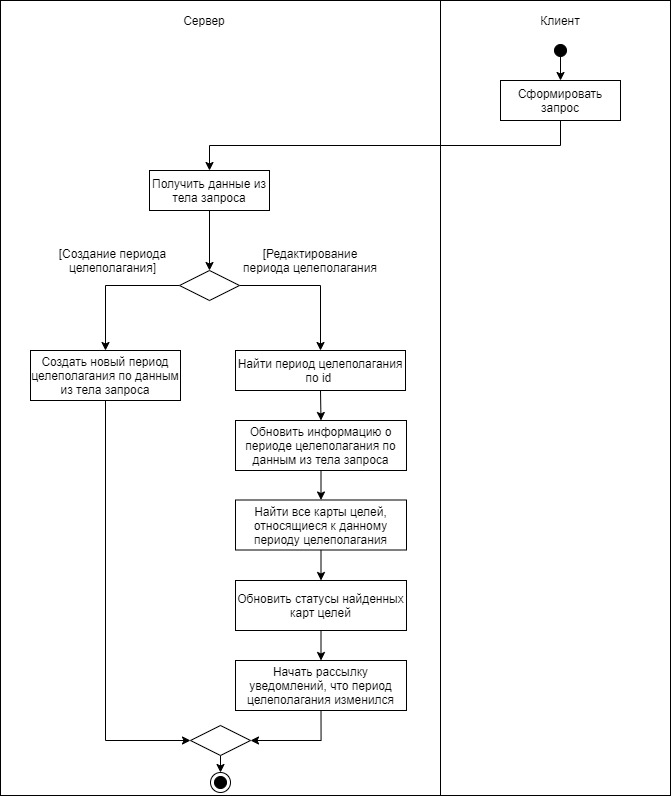


Рисунок 2.10 – Алгоритм создания и редактирования периода целеполагания

В клиентской части приложения формируется запрос на сервер. Тело запроса имеет структуру, представленную в листинге 2.1.

Листинг 2.1 – Структура тела запроса

{

"id": "6971234231179996598",

"status\_id": "6973896685878866189",

"gs\_year": 2022,

"last\_date\_employment": "2022-05-02T00:00:00.000Z",

"last\_date\_transfer": "2022-05-13T00:00:00.000Z"

}

Если параметр id в теле запроса не пустой, то происходит редактирование периода целеполагания, иначе, создание. При редактировании периода целеполагания алгоритм обновляет статусы карт целей, которые относятся к нему, инициирует рассылку уведомлений о смене этого статуса.

Проектирование алгоритма создания карты целей

Алгоритм создание карты целей представлен в виде диаграммы деятельности на рисунке 2.11.

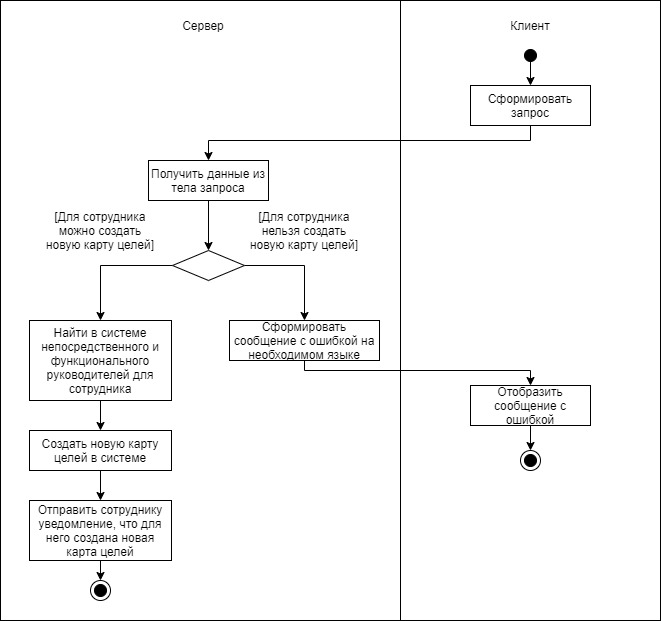


Рисунок 2.11 – Алгоритм создания карты целей

В клиентской части приложения формируется запрос на сервер. Тело запроса имеет структуру, представленную в листинге 2.2.

Листинг 2.2 – Структура тела запроса для создания карты целей

{

"position\_ids": "7073090170805933793",

"period\_id": "7021104036976751350"

}

Серверная часть приложения проверяет доступность создания новой карты цели для сотрудника. Если создание недоступно, то пользователю отображается всплывающее окно с сообщением об ошибке. Окно с ошибкой на русском и английском языках представлено на рисунке 2.12.

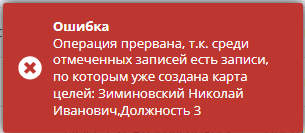


Рисунок 2.12 – Всплывающее окно с ошибкой

Если создание карты целей доступно, то в системе находятся непосредственный и функциональный руководители пользователя, на основе данных из тела запроса и данных о руководителях создается новая карта целей. Сотруднику, для которого она создаются отправляется соответствующее уведомление.

Также при создании нового периода целеполагания предусмотрено автоматическое создания карт целей.

Проектирование алгоритма создания и редактирования цели

Алгоритм создания и редактирование цели представлен в виде диаграммы деятельности на рисунке 2.13.

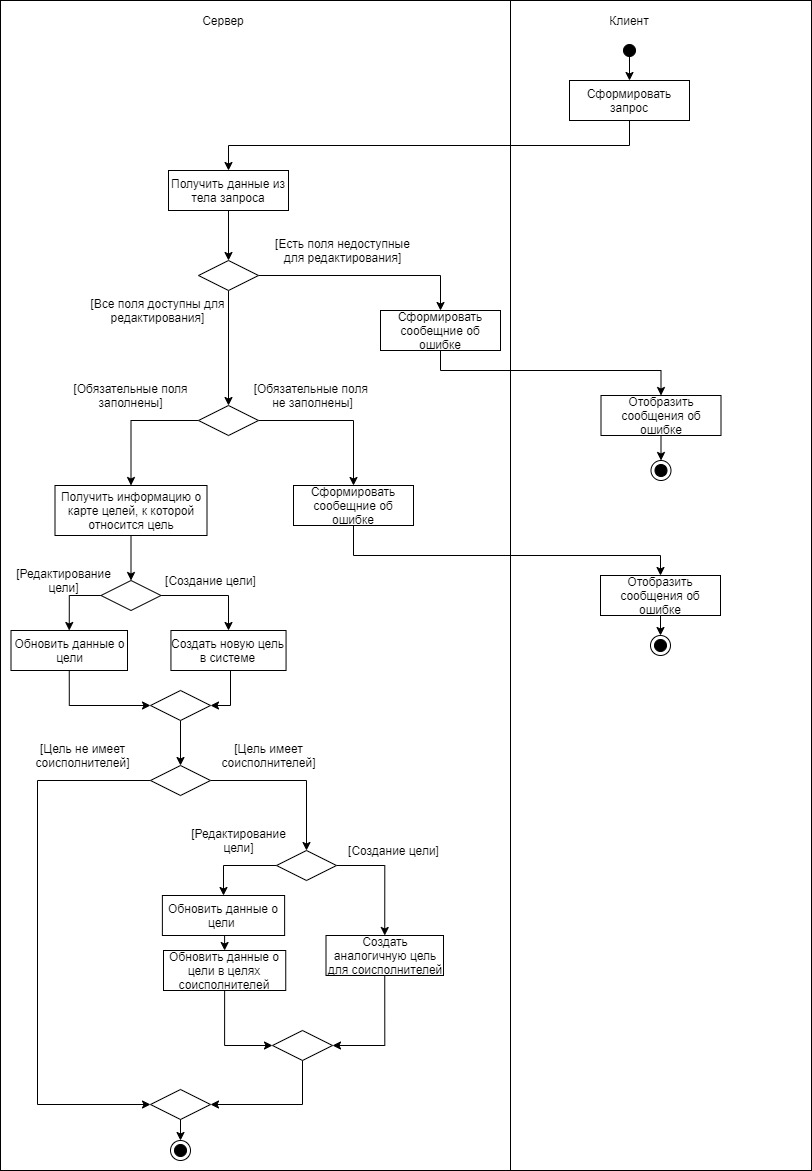


Рисунок 2.13 – Алгоритм создания и редактирования цели

По данному алгоритмы клиентская часть приложения формирует запрос. Серверная часть приложения получает данные из тела запроса. Если тело запроса содержит параметр id, то происходит редактирование цели, иначе – создание.

Если изменились поля, которые недоступны для редактирования, то отображается сообщение об ошибке.

Если есть поля, которые обязательны для заполнения, но они не заполнены, также отображается ошибка.

Далее алгоритм получает информацию о карте цели сотрудника, создается новая цель или обновляются данные существующей. Если цель имеет соисполнителей, то соответственно происходит создание или редактирования цели для соисполнителей. При добавлении новой цели автоматически выстраивается иерархия целей. Алгоритм определения номера цели представлен в виде диаграммы активности на рисунке 2.14.

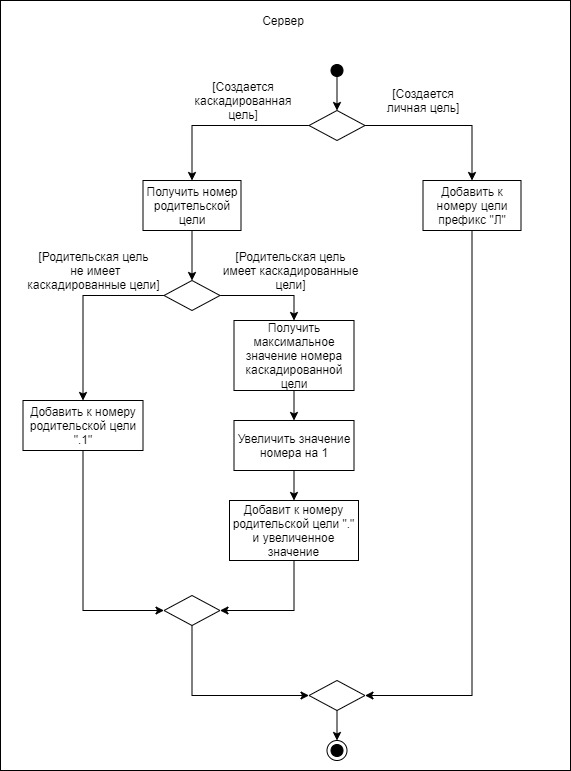


Рисунок 2.14 – Алгоритм определения номера цели

Проектирование алгоритмов переключения языка

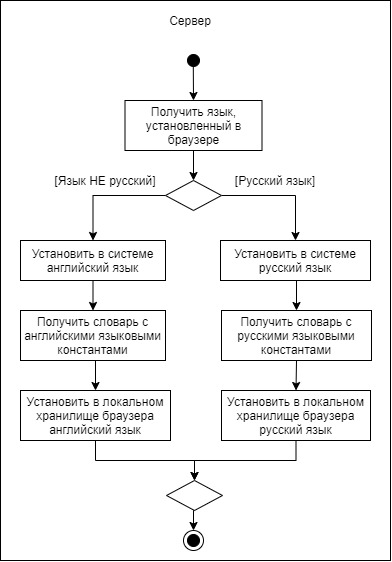
Алгоритм определения языка при первом открытии портала представлен в виде диаграммы деятельности на рисунке 2.15.

Рисунок 2.15 – Алгоритм определения языка при первом открытии портала

При первом открытии портала определяется язык, установленный в браузере пользователя. Если язык отличен от русского, то в системе устанавливается английский язык, происходит получение словаря с английскими языковыми константами, в локальном хранилище браузера, иначе аналогичные действия с русским языком.

Алгоритм переключения языка представлен в виде диаграммы деятельности на рисунке 2.16.

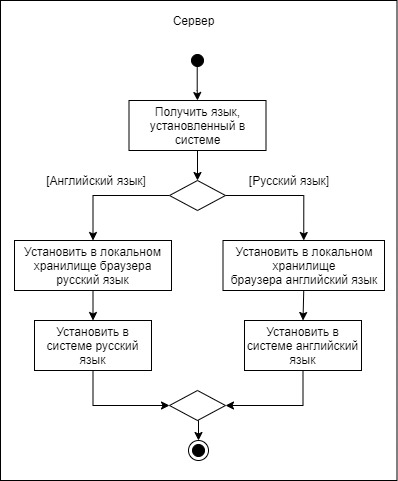


Рисунок 2.16 – Алгоритм переключения языка

Проектирование алгоритма построения отчетной документации

Алгоритм построения отчетной документации представлен в виде диаграммы деятельности на рисунке 2.17.

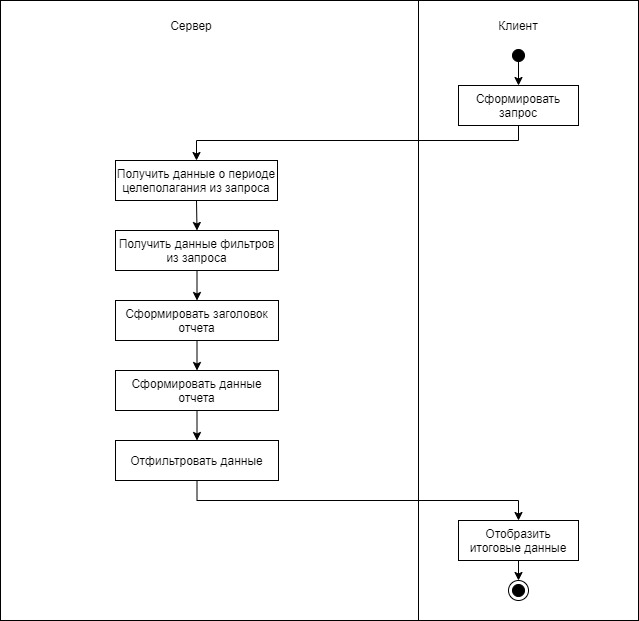


Рисунок 2.17 – Алгоритм построения отчетной документации

Клиентская часть приложения формирует запрос на сервер для получения отчетной документации. Запрос содержит данные о периоде целеполагания и фильтры.

Серверная часть приложения формирует заголовок отчета, а также данные о нем, после чего они фильтруются. Итоговые данные отображаются в таблице в клиентской части приложения.

## Выводы по разделу

В данном разделе спроектирован бизнес-процесс автоматизации оценки и анализа ключевых показателей производительности работы сотрудников.

Спроектированы экранные формы:

* работа с периодами целеполагания;
* управление картами целей;
* управление картой цели сотрудника;
* история согласования карты цели;
* отчетная документация.

Доступ к экранным формам разграничен по ролям.

Спроектирована база данных программной системы в виде ER‑диаграммы, выполненной в нотации Джеймса Мартина.

Выбраны средства разработки программной системы.

Спроектированы алгоритмы работы программной системы:

* создание и редактирования периода целеполагания;
* создание карты целей;
* создание и редактирование целей;
* переключение языка;
* построение отчетной документации.

# Технологический раздел

## Настройка окружения разработки

Разработка производилась в редакторе исходного кода Visual Studio Code. Данный редактор предназначен для кроссплатформенной разработки веб- и облачных решений, распространяется под свободной лицензией MIT.

Установка зависимостей

Для разработки клиентской части программной системы установлены следующие зависимости:

* ngx-translate – библиотека интернационализации приложений;
* devextreme – коллекция высокопроизводительных и отзывчивых виджетов пользовательского интерфейса[7];
* ngx-toastr – виджет пользовательского интерфейса, предназначенный для настройки и кастомизации уведомлений;
* rxjs – библиотека для создания асинхронных и событийных программ с использованием наблюдаемых последовательностей;
* date-fns – библиотека для работы с датами;
* material – библиотека UI-компонентов.

После установки зависимостей формируется файл с названием «package.json», представленный на в листинге 3.1.

Листинг 3.1 – Файл «package.json»

{

"name": " webtutor-kpi",

"version": "0.0.14",

"scripts": {

"ng": "ng",

"start": "concurrently --kill-others \"ng serve\" \"node ./cors/index.js\"",

"build": "ng build --aot true --build-optimizer false --common-chunk false --optimization true --output-hashing all --base-href ./",

"publish": "node publish.js",

"publishRelease": "node publishRelease.js",

"buildLocal": "node date.js && ng build --base-href ./ && node publishLocal.js",

"tag": "node saveTag.js",

"watch": "ng build --watch --configuration development",

"test": "ng test"

},

Продолжение листинга 3.1

"private": true,

"dependencies": {

"@angular/animations": "~12.0.4",

"@angular/cdk": "~12.0.4",

"@angular/common": "~12.0.4",

"@angular/compiler": "~12.0.4",

"@angular/core": "~12.0.4",

"@angular/forms": "~12.0.4",

"@angular/material": "~12.0.4",

"@angular/platform-browser": "~12.0.4",

"@angular/platform-browser-dynamic": "~12.0.4",

"@angular/router": "~12.0.4",

"@ngx-translate/core": "^13.0.0",

"@ngx-translate/http-loader": "^6.0.0",

"@popperjs/core": "^2.9.2",

"@tinkoff/utils": "^2.2.1",

"angular-super-validator": "^2.0.0",

"class-transformer": "^0.4.0",

"date-fns": "^2.22.1",

"devextreme": "21.1.5",

"devextreme-angular": "21.1.5",

"devextreme-schematics": "^1.2.15",

"iframe-resizer": "^4.3.2",

"ngx-cookie-service": "^12.0.3",

"ngx-popperjs": "^11.0.0",

"ngx-toastr": "^14.0.0",

"reflect-metadata": "^0.1.13",

"rxjs": "~6.6.0",

"tslib": "^2.1.0",

"zone.js": "~0.11.4"

},

"devDependencies": {

"@angular-devkit/build-angular": "~12.0.4",

"@angular/cli": "~12.0.4",

"@angular/compiler-cli": "~12.0.4",

"@types/node": "^12.11.1",

"azure-devops-node-api": "^11.1.0",

"body-parser": "^1.19.0",

"concurrently": "^6.2.0",

"cookie-parser": "^1.4.5",

"express": "^4.17.1",

"fs-extra": "^10.0.0",

"moment": "^2.29.1",

"request": "^2.88.2",

"typescript": "~4.2.3"

}

}

Настройка форматирования кода с помощью Prettier

Prettier – инструмент для форматирования кода в соответствии с определенными правилами. Данный инструмент позволяет автоматически поддерживать целостность кода проекта. Для переопределения стандартных настроек Prettier необходимо добавить в корневую папку проекта файл «.prettierrc.json». Файл настроек для проекта представлен в листинге 3.2.

Листинг 3.2 – Файл «.prettierrc.json»

{

"tabWidth": 2,

"useTabs": true,

"singleQuote": true,

"semi": true,

"bracketSpacing": true,

"arrowParens": "always",

"trailingComma": "all",

}

В данном файле описаны следующие настройки форматирования кода:

* tabWidth – указывает, сколько пробелов должно использоваться при табуляции;
* useTabs – использовать символ табуляции для отступов;
* singleQuote – использовать одинарные кавычки для обычных строк в коде;
* semi – указывает, что каждая строка должна оканчиваться знаком точки с запятой;
* bracketSpacing – печатать пробелы между скобками в литералах объекта;
* arrowParens – использовать круглые при единственном параметре в стрелочных функциях;
* trailingComma – печатать конечные запятые в многострочных синтаксических конструкциях, разделенных запятыми;

## Реализация пользовательских интерфейсов

Все интерфейсы реализованы в виде Angular-компонентов. Каждый компонент состоит из трех файлов:

* файл HTML разметки;
* TypeScript файл, который содержит логику работу;
* файл стилей SCSS.

Элементами HTML разметки являются тэги.

Реализация интерфейса для работы с периодами целеполагания

Логика работы интерфейса работы с периодами целеполагания реализована в классе PeriodListComponent. Данный класс представлен в листинге 3.3.

Листинг 3.3 – Класс PeriodListComponent

@Component({

selector: 'app-period-list',

templateUrl: './period-list.component.html',

styleUrls: ['./period-list.component.scss'],

})

export class PeriodListComponent implements OnInit {

@ViewChild('dataGrid') dataGrid!: DxDataGridComponent;

dataSource!: any;

statuses: PeriodsStatusModel[] = [];

updateListSubject: Subject<any> = new Subject<any>();

yearValidationPattern = /^[0-9]{4}$/;

create() {

const dialogRef = this.dialog.open(CreateEditPeriodDialogComponent, {

disableClose: true,

width: '800px',

height: '350px',

panelClass: 'add-goal-dialog-container',

data: null,

});

dialogRef.beforeClosed().subscribe((result) => {

if (result) {

this.updateListSubject.next();

}

});

}

delete() {

return {

click: ($event: any) => {

const dataHelperWithCallback: HelperWithCallbackDialogData = {

Продолжение листинга 3.3

text: this.deleteMessageTemplate.replace('{0}',$event.row.data.gs\_year)

}

const dialogRef this.dialog.open(HelperWithCallbackDialogComponent, {

disableClose: true,

data: dataHelperWithCallback

});

dialogRef.beforeClosed().subscribe((result: boolean) => {

if (result) {

this.periodService

.delete($event.row.data.id)

.toPromise()

.then((\_) => {

this.updateListSubject.next();

this.dataGrid.instance.refresh(true);

});

}

});

},

};

}

edit() {

return {

click: ($event: any) => {

const dialogRef = this.dialog.open(CreateEditPeriodDialogComponent, {

disableClose: true,

width: '800px',

height: '350px',

data: $event.row.data,

});

dialogRef.afterClosed().subscribe((result: boolean) => {

if (result) {

this.updateListSubject.next();

this.dataGrid.instance.refresh();

}

});

},

};

}

}

Данный класс имеет декоратор «@Component», поэтому представляет собой компонент Angular.

Описание полей класса представлено в таблице 3.1.

Таблица 3.1 – Описание полей класса PeriodListComponent

|  |  |
| --- | --- |
| Поле | Описание |
| dataGrid | Данное поле хранит в себе ссылку на таблицу |
| dataSource | Источник данных периодов целеполагания |
| statuses | Коллекция статусов периодов целеполагания |

Продолжение таблицы 3.1

|  |  |
| --- | --- |
| Поле | Описание |
| updateListSubject | Поле, которое позволяет автоматически обновлять данные о периодах целеполагания |
| yearValidationPattern | Регулярное выражения для проверки введенного значения года |

В таблице 3.2 представлено описание методов класса PeriodListComponent.

Таблица 3.2 – Описание методов класса PeriodListComponent

|  |  |
| --- | --- |
| Метод | Описание |
| create | Метод, отвечающий за создание нового периода целеполагания. Данный метод открывает диалоговое окно, в котором пользователь должен заполнить необходимые поля и нажать на кнопку «Сохранить». После нажатия кнопки отсылается запрос на сервер на добавление нового периода целеполагания. |
| delete | Метод отвечает за удаление периода целеполагания. Данный метод открывает диалоговое окно с подтверждением или отменой удаления. Если нажата кнопка «Да», то на сервер отсылается запрос на удаление периода целеполагания, иначе диалоговое окно закрывается |
| edit | Метод отвечает за редактирование периода целеполагания. Открывается диалоговое окно аналогичное добавлению, но с уже заполненными полями. В данном окне пользователь может изменить значения в полях и нажать на кнопку «Сохранить», после этого на сервер отсылается запрос на редактирование периода целеполагания. |

Файл HTML разметки представлен в листинге 3.4.

Листинг 3.4 – HTML разметка страницы для работы с периодами целеполагания

<header>

<span class="add-period">

<dx-button

type="default"

text="{{ 'periods.createPeriod' | translate}}"(onClick)="create()">

</dx-button>

<button

mat-icon-button

[attr.aria-label]="'general.exportMapListToExcel' | translate"

matTooltip="{{ 'general.exportMapListToExcel' | translate }}"

(click)="exportPeriodToExcel()"

>

<mat-icon>system\_update\_alt</mat-icon>

</button>

</span>

</header>

<content>

<dx-data-grid

#dataGrid

class="new-design period-grid"

[dataSource]="dataSource"

[showBorders]="false"

[columnAutoWidth]="true"

>

<dxo-load-panel [enabled]="false"></dxo-load-panel>

<dxo-editing

[allowAdding]="false"

[allowUpdating]="false"

[allowDeleting]="false"

>

</dxo-editing>

<dxo-state-storing [enabled]="true" type="localStorage"

storageKey="periodListGridStorage"></dxo-state-storing>

<dxo-filter-row [visible]="true"></dxo-filter-row>

<dxi-column alignment="center" dataField="gs\_year"

width="100"

[allowEditing]="false" caption="{{ 'periods.year' |

translate }}">

</dxi-column>

<dxi-column

dataField="status\_id"

minWidth="200"

caption="{{ 'periods.periodStatus' | translate }}"

>

<dxo-lookup

[dataSource]="statuses"

valueExpr="id"

displayExpr="name"

>

</dxo-lookup>

</dxi-column>

<dxi-column dataField="last\_date\_employment"

[allowEditing]="false"

alignment="center"

width="200"

caption="{{ 'periods.deadlineForAdmission' | translate }}">

</dxi-column>

<dxi-column dataField="last\_date\_transfer"

[allowEditing]="false"

Продолжение листинга 3.4

alignment="center"

width="200"

caption="{{ 'periods.deadlineForTranslation' | translate }}">

</dxi-column>

<dxi-column type="buttons" width="100">

<dxi-button icon="edit" [onClick]="edit().click"></dxi-button>

<dxi-button

icon="-"

template="showMenuUnfinishedToWorkButton"

>

</dxi-button>

<dxi-button icon="trash" [onClick]="delete().click"></dxi-button>

</dxi-column>

<a \*dxTemplate="let cellData of 'showMenuUnfinishedToWorkButton'">

<mat-icon

[matMenuTriggerFor]="showMenuUnfinishedToWork"

class="show-menu"

>

group

</mat-icon>

<mat-menu #showMenuUnfinishedToWork="matMenu" class="width-mat-menu">

<button

mat-menu-item

(click)="showOnClick(cellData.data.id, 1)"

>

{{ "periods.showUnfinishedWorkWithOrder1" | translate }}

</button>

<button

mat-menu-item

(click)="showOnClick(cellData.data.id, 2)"

>

{{ "periods.showUnfinishedWorkWithOrder2" | translate }}

</button>

<button

mat-menu-item

(click)="showOnClick(cellData.data.id, 3)"

>

{{ "periods.showUnfinishedWorkWithFact" | translate }}

</button>

</mat-menu>

</a>

</dx-data-grid>

</content>

Разработанная страница работы с периодами целеполагания представлена на рисунке 3.1

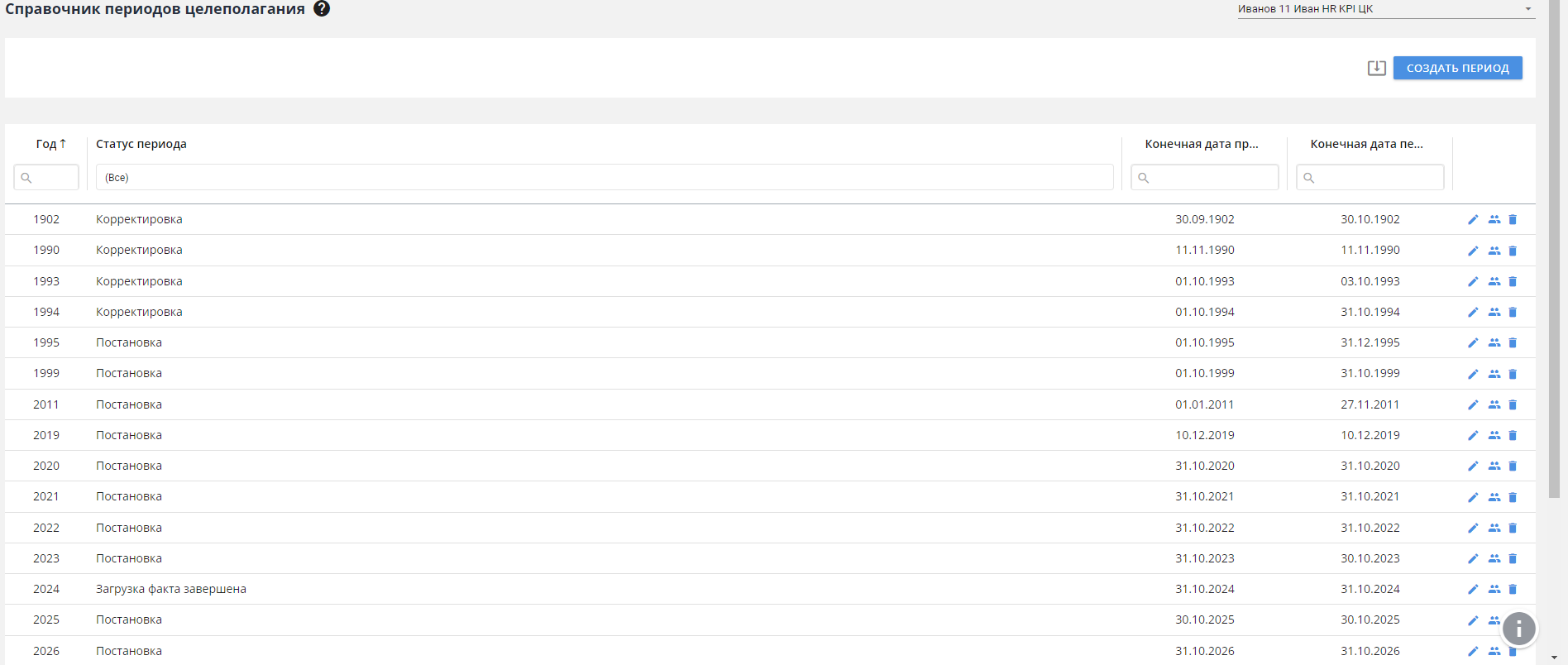


Рисунок 3.1 – Страница работы с периодами целеполагания

Диалоговое окно редактирования и создания периода целеполагания представлено на рисунке 3.2.

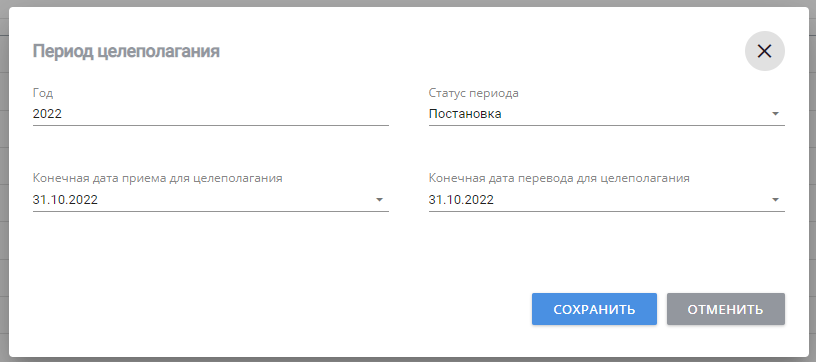


Рисунок 3.2 – Диалоговое окно редактирования и создания периода целеполагания

Диалоговое окно подтверждения удаления периода целеполагания представлено на рисунке 3.3.

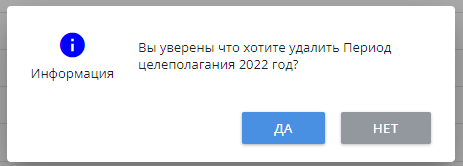


Рисунок 3.3 – Диалоговое окно подтверждения удаления периода целеполагания

Таким образом разработан Angular-компонент PeriodListComponent, предназначенный для управления периодами целеполагания.

Реализация интерфейса управления картами целей

Логика работы интерфейса управления картами целей реализована в классе MapListComponent. Данный класс представлен в листинге 3.5.

Листинг 3.5 – Класс MapListComponent

@Component({

selector: 'app-map-list',

templateUrl: './map-list.component.html',

styleUrls: ['./map-list.component.scss'],

})

export class MapListComponent {

@ViewChild('dataGrid') dataGrid!: DxDataGridComponent;

periods: PeriodSelectModel[] = [];

statusesDictionary: GoalsMapStatModel[] = [];

treeData: OrgModel[] = [];

dataSource: DataSource;

showLeftBoxColumn = true;

gridInitialized = false;

loadDataSubject = new Subject();

showDissmised: boolean = false;

Продолжение листинга 3.5

async create() {

const positionIds = this.positionIdsCollection.map(item => item.position\_id).join(',');

this.mainService.createGoalMap(positionIds, this.periodValue).subscribe(async () => {

await this.updateGridData();

});

}

async copy() {

const dataCreatePredecessorMapsDialog: CreatePredecessorMapsDialogData = {

period: this.periodValue,

positions: this.positionIdsCollection

}

const dialogRef = this.dialog.open(CreatePredecessorMapsDialogComponent, {

disableClose: true,

panelClass: 'add-goal-dialog-container',

data: dataCreatePredecessorMapsDialog

});

dialogRef.afterClosed().subscribe(async (result: boolean) => {

if (result) {

await this.updateGridData(); // обновляем данные грида

}

});

}

edit() {

const mapIds = this.positionIdsCollection.map(item => item.map\_id).join(',');

const dialogRef = this.dialog.open(GoalMapsStatusesComponent, {

disableClose: true,

width: '600px',

data: {

mapIds

}

});

dialogRef.afterClosed().subscribe((result: any) => {

const { status, value } = result;

if (status) {

this.goalsListService.setGoalMapsStatuses(mapIds, value.status\_id).subscribe(\_ => {

this.loadDataSubject.next()

});

}

});

}

delete() {

const ids = this.positionIdsCollection.map(item => item.map\_id);

this.mainService.deleteGoalMap(ids.join()).subscribe(() => {

this.dataGrid.instance.deselectAll(); // чтобы не запоминались выбранные чекбоксы, кажется они попадают в state грида

// noinspection JSIgnoredPromiseFromCall

this.dataGrid.instance.refresh();

});

}

cancel() {

const ids = this.positionIdsCollection.map(item => item.map\_id);

this.mainService.cancelGoalMap(ids.join()).subscribe(() => {

Продолжение листинга 3.5

// noinspection JSIgnoredPromiseFromCall

this.dataGrid.instance.deselectAll();

// noinspection JSIgnoredPromiseFromCall

this.dataGrid.instance.refresh();

});

}

async sendReminder(type: number) {

await this.mainService.sendReminders(type, this.periodValue).toPromise()

this.popover?.instance.hide()

}

}

Данный класс имеет декоратор «@Component», поэтому представляет собой компонент Angular.

Описание полей класса MapListComponent представлено в таблице 3.4.

Таблица 3.4 – Описание полей класса MapListComponent

|  |  |
| --- | --- |
| Поле | Описание |
| dataGrid | Данное поле хранит в себе ссылку на таблицу с сотрудниками и их картами целей |
| dataSource | Источник данных карт целей |
| treeData | Дерево подразделений |
| loadDataSubject | Поле, которое позволяет автоматически обновлять данные о картах целей |
| showDissmised | Флаг, отвечающий за показ карт целей уволенных сотрудников |
| periods | Коллекция периодов целеполагания |
| statusesDictionary | Коллекция статусов карт целей |

Описание методов класса MapListComponent представлено в таблице 3.5.

Таблица 3.5 – Описание методов класса MapListComponent

|  |  |
| --- | --- |
| Метод | Описание |
| create | Метод вызываюется по нажатию на кнопку «Добавить». Отправляет запрос на сервер с выбранными сотрудникам для добавления им новых карт целей |
| copy | По нажатию на кнопку «Создать на основе предшественника» открывает диалоговое окно, в котором необходимо выбрать на основе чьей карты целей необходимо создать новую |
| edit | Метод, отвечающий за редактирование статуса карты целей. По нажатию на кнопку «Редактировать» открывается диалоговое окно, в котором пользователь может поменять статус выбранным картам целей |
| cancel | Метод, отвечающий за отмену карт целей |
| delete | Метод, отвечающий за удаление карт целей |
| sendReminder | Метод, отвечающий за отправку шаблонных уведомлений: по текущему периоду целеполагания, об окончании периода целеполагания, руководителям о картах, находящихся на согласовании |
| addAlert | Метод, вызывающий диалоговое окно для отправки уведомлений выбранным сотрудникам. В диалоговом окне можно задать тему и текст уведомления |

Файл HTML разметки представлен в листинге 3.5.

Листинг 3.6 – HTML разметка страницы управления картами целей

<div style="display: flex; align-items: center">

<div class="select-wrapper">

Продолжение листинга 3.6

<selectable-period-from-list

(changePeriod)="changePeriod($event)"

></selectable-period-from-list>

</div>

</div>

<div \*ngIf="gridInitialized" class="grid-controls small">

<div style="display: flex">

<div class="checkbox-control">

<dx-check-box (onValueChanged)="onShowRecursiveValueChanged($event)" [value]="showRecursive"

class="checkbox-element" text="{{'maps.showRecursive' | translate}}"></dx-check-box>

</div>

<div class="checkbox-control">

<dx-check-box (onValueChanged)="onShowCompletedPositions($event)" [value]="showDissmised"

class="checkbox-element" text="{{'main.showCompletedPositions' | translate}}"></dx-check-box>

</div>

</div>

<div style="display: flex">

<div

#tooltip="matTooltip"

[matTooltipDisabled]="!checkManagementButtonsStates().add"

class="grid-control"

matTooltip="{{ 'general.add' | translate }}"

>

<button

(click)="create()"

[disabled]="checkManagementButtonsStates().add"

mat-icon-button>

<mat-icon>note\_add</mat-icon>

</button>

</div>

<div

#tooltip="matTooltip"

[matTooltipDisabled]="!checkManagementButtonsStates().copy"

class="grid-control"

matTooltip="{{ 'general.copy' | translate }}"

>

<button

(click)="copy()"

[disabled]="checkManagementButtonsStates().copy"

mat-icon-button>

<mat-icon>file\_copy</mat-icon>

</button>

</div>

<div

#tooltip="matTooltip"

[matTooltipDisabled]="!checkManagementButtonsStates().statusChange"

class="grid-control"

matTooltip="{{ 'maps.edit' | translate }}"

>

<button

(click)="edit()"

[disabled]="checkManagementButtonsStates().statusChange"

mat-icon-button>

<mat-icon>edit</mat-icon>

</button>

</div>

Продолжение листинга 3.6

<div

#tooltip="matTooltip"

[matTooltipDisabled]="!checkManagementButtonsStates().delete"

class="grid-control"

matTooltip="{{ 'maps.delete' | translate }}"

>

<button

(click)="delete()"

[disabled]="checkManagementButtonsStates().delete"

mat-icon-button>

<mat-icon>delete</mat-icon>

</button>

</div>

<div

#tooltip="matTooltip"

[matTooltipDisabled]="!checkManagementButtonsStates().cancel"

class="grid-control"

matTooltip="{{ 'maps.cancel' | translate }}"

>

<button

(click)="cancel()"

[disabled]="checkManagementButtonsStates().cancel"

mat-icon-button>

<mat-icon>cancel</mat-icon>

</button>

</div>

<div

#tooltip="matTooltip"

[matTooltipDisabled]="!checkManagementButtonsStates().addAlert"

class="grid-control"

matTooltip="{{ 'general.sendNotification' | translate }}"

>

<button

(click)="addAlert()"

[disabled]="checkManagementButtonsStates().addAlert"

mat-icon-button>

<mat-icon>add\_alert</mat-icon>

</button>

</div>

</div>

</div>

<div style="display: flex">

<div class="grid-control">

<borderButtonWrapper (onClick)="create()" [disabled]="checkManagementButtonsStates().add"

[matTooltipText]="'general.add' | translate">

<mat-icon>note\_add</mat-icon>

{{'general.add' | translate}}

</borderButtonWrapper>

</div>

<div class="grid-control">

<borderButtonWrapper (onClick)="copy()" [disabled]="checkManagementButtonsStates().copy"

[matTooltipText]="'general.copy' | translate">

<mat-icon>file\_copy</mat-icon>

{{'general.copy' | translate}}

</borderButtonWrapper>

</div>

<div class="grid-control">

Продолжение листинга 3.6

<borderButtonWrapper (onClick)="edit()" [disabled]="checkManagementButtonsStates().statusChange"

[matTooltipText]="'general.edit' | translate">

<mat-icon>edit</mat-icon>

{{'maps.edit' | translate}}

</borderButtonWrapper>

</div>

<div class="grid-control">

<borderButtonWrapper (onClick)="delete()" [disabled]="checkManagementButtonsStates().delete"

[matTooltipText]="'maps.delete' | translate">

<mat-icon>delete</mat-icon>

{{'maps.delete' | translate}}

</borderButtonWrapper>

</div>

<div class="grid-control">

<borderButtonWrapper (onClick)="cancel()" [disabled]="checkManagementButtonsStates().cancel"

[matTooltipText]="'maps.cancel' | translate">

<mat-icon>cancel</mat-icon>

{{'maps.cancel' | translate}}

</borderButtonWrapper>

</div>

<div class="grid-control">

<borderButtonWrapper (onClick)="addAlert()"

[disabled]="checkManagementButtonsStates().addAlert"

[matTooltipText]="'maps.sendNotification' | translate">

<mat-icon>add\_alert</mat-icon>

{{'maps.sendNotification' | translate}}

</borderButtonWrapper>

</div>

<content>

<dx-data-grid

#dataGrid

(onRowPrepared)="preparedRow($event)"

(onSelectionChanged)="onGridSelectionChanged($event)"

[activeStateEnabled]="true"

[allowColumnReordering]="true"

[allowColumnResizing]="true"

[columnMinWidth]="100"

[dataSource]="dataSource"

[remoteOperations]="{ paging: true, filtering: true, sorting: true }"

[showBorders]="false"

[style.marginLeft.px]="isDrawerOpen ? 340 : 32"

class="staff-grid new-design"

height="calc(100vh - 220px)"

width="calc(100% - {{isDrawerOpen?350:48}}px)"

>

<dxo-load-panel [enabled]="false"></dxo-load-panel>

<dxo-scrolling rowRenderingMode="virtual"></dxo-scrolling>

<dxo-paging [enabled]="true" [pageSize]="10"></dxo-paging>

<dxo-pager

[allowedPageSizes]="[5, 10, 20, 100]"

[displayMode]="'full'"

[showInfo]="true"

[showNavigationButtons]="true"

[showPageSizeSelector]="true"

[visible]="true"

>

Продолжение листинга 3.6

</dxo-pager>

<dxo-state-storing [enabled]="true" storageKey="staffGridStorage" type="localStorage"></dxo-state-storing>

<dxo-selection mode="multiple" showCheckBoxesMode="always"></dxo-selection>

<dxo-filter-row [visible]="true"></dxo-filter-row>

<dxi-column [width]="200" caption="{{ 'maps.fullName' | translate }}" cellTemplate="shortName" cssClass="style-fullname" dataField="fullname"></dxi-column>

<div \*dxTemplate="let cellData of 'shortName'">

<div style="display: inline-flex; justify-content: space-between; width: 100%; align-items: center">

<div>

</div>

<dxi-column [width]="200" caption="{{ 'maps.position' | translate }}" dataField="position\_name"></dxi-column>

<dxi-column [width]="200" caption="{{ 'maps.stage' | translate }}" dataField="stage"></dxi-column>

<dxi-column [width]="200" caption="{{ 'maps.status' | translate }}" dataField="map\_status\_name">

<dxo-lookup

[dataSource]="statusesDictionary"

displayExpr="name"

valueExpr="name">

<dxo-drop-down-options

[closeOnOutsideClick]="true"

[showTitle]="false">

</dxo-drop-down-options>

</dxo-lookup>

</dxi-column>

</dx-data-grid>

</content>

</base-page-layout>

Разработанная страница управления картами целей представлена на рисунке 3.5.

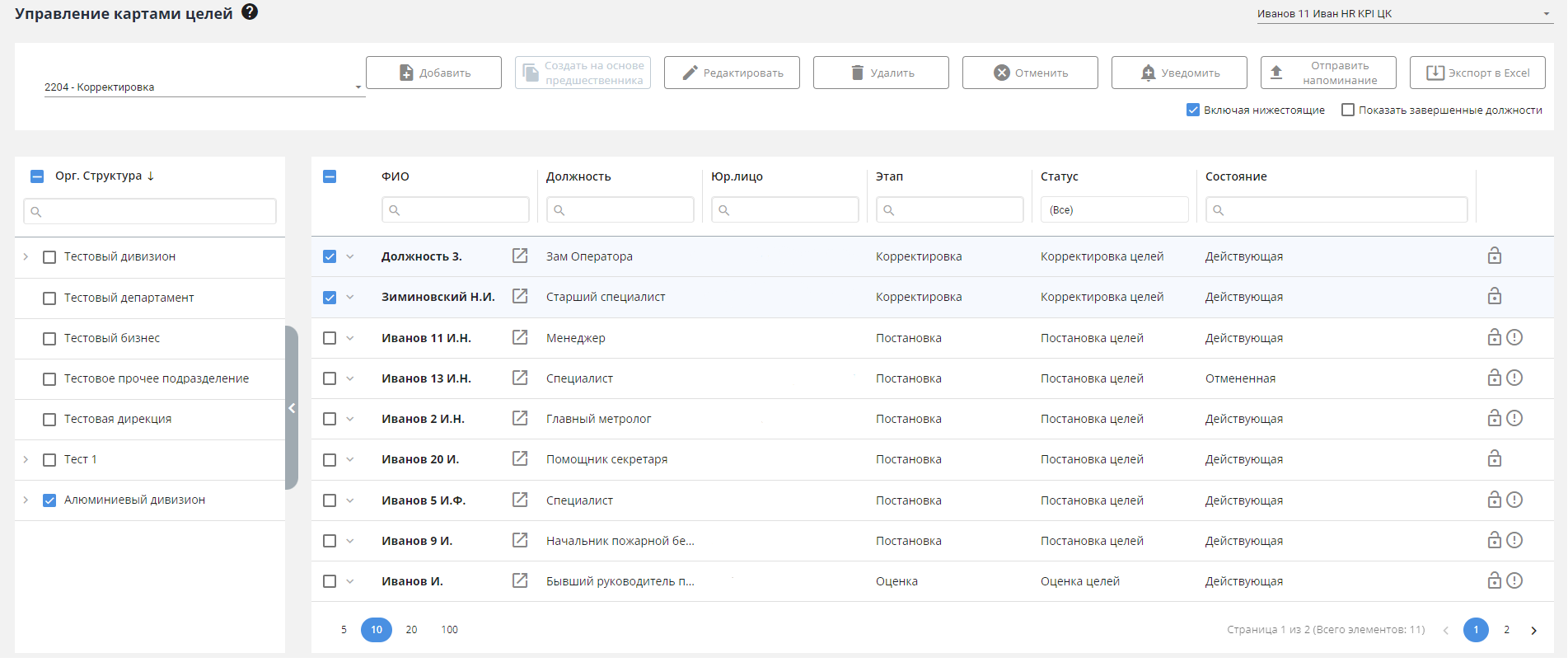


Рисунок 3.5 – Страница управления картами целей

Диалоговое окно создания карты целей на основе предшественника представлено на рисунке 3.6.

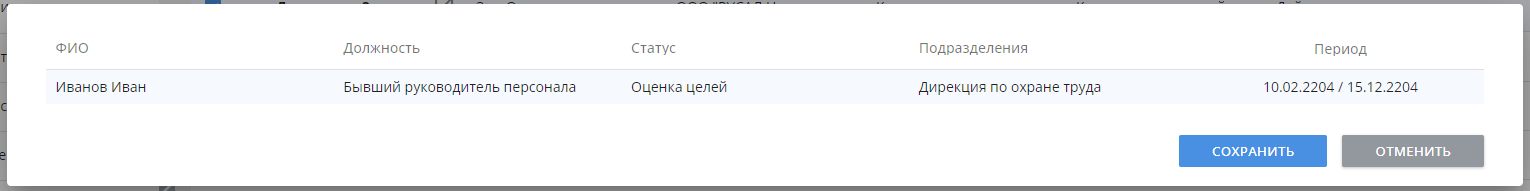


Рисунок 3.6 – Диалоговое окно создания карты целей на основе предшественника

Диалоговое окно редактирования статуса карты целей представлено на рисунке 3.7.

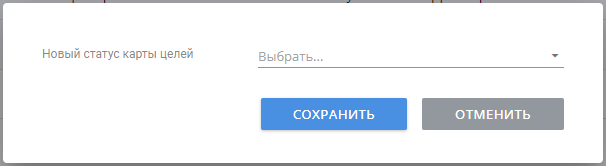


Рисунок 3.7 – Диалоговое окно редактирования статуса карты целей

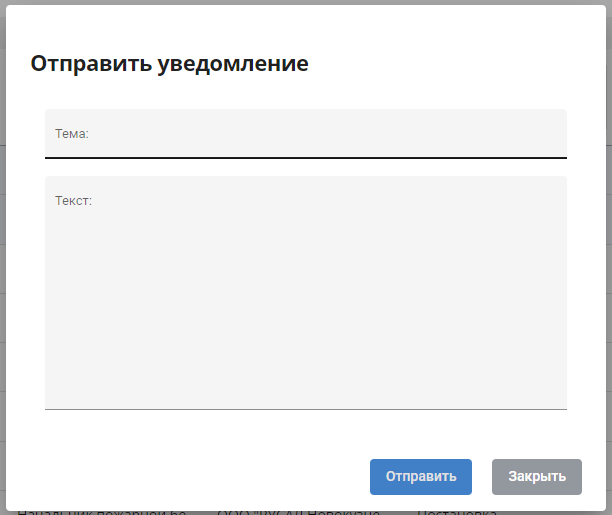
Диалоговое окно отправки уведомлений представлено на рисунке 3.8.

Рисунок 3.8 – Диалоговое окно для отправки уведомлений

Таким образом разработан Angular-компонент MapListComponent необходимый для управления картами целей сотрудников.

Реализация интерфейса управления картой цели сотрудника

Логика работы интерфейса управления картой цели сотрудника реализована в классе MainViewComponent. Данный класс представлен в листинге 3.7.

Листинг 3.7 – Класс MainViewComponent

@Component({

selector: 'app-main-view',

templateUrl: './main-view.component.html',

styleUrls: ['./main-view.component.scss'],

})

export class MainViewComponent implements OnInit {

@ViewChild('dataGrid') dataGrid!: DxDataGridComponent;

baseInfo$ = new Subject()

isShowCanceled = false;

isGoalsWithZeroWeight = true;

goalInfo: GoalsParentModel = new GoalsParentModel();

statusesDictionary: GoalsMapStatModel[] = [];

addCascading(goal: any, $event: any) {

$event.stopPropagation();

let user = new GoalsMapModel();

user.position\_id = this.baseInfo.position\_id;

user.position\_name = this.baseInfo.position\_name;

user.fullname = this.baseInfo.fullname;

user.subdivision\_name = this.baseInfo.division\_name;

const dataCreateEditGoalView: CreateEditGoalViewData = {

editModel: goal,

map: this.paramId,

period: this.baseInfo.period\_id,

isCascading: true,

isNotEditable: false,

user: user

}

const dialogRef = this.dialog.open(CreateEditGoalViewComponent, {

disableClose: true,

width: '900px',

height: '965px',

panelClass: 'add-goal-dialog-container',

data: dataCreateEditGoalView,

});

dialogRef.afterClosed().subscribe((result: boolean) => {

if (result) {

this.reloadGoals();

Продолжение листинга 3.7

}

});

}

cancelWithComment(goal: any) {

const dataChangeStatusModal: ChangeStatusModalData = {

width: '500px',

text: this.translateService.instant('main.item.reasonForRejectingTheGoal'),

isCommentRequired: true

}

const dialogCancelCommentRef = this.dialog.open(ChangeStatusModalComponent, {

disableClose: true,

data: dataChangeStatusModal

});

dialogCancelCommentRef.afterClosed().subscribe((comment: string) => {

this.mainService.cancelGoal(goal.id, comment).subscribe(() => {

this.reloadGoals(goal.kind\_id === KindEnum.kindCrossId);

})

})

return

}

callCanceling(goal: any, $event: any) {

$event.stopPropagation();

this.mainService.checkGoalCancel(goal.id).subscribe((data) => {

const {cascadedGoalExist, showComment} = data;

if (cascadedGoalExist) {

const data: HelperWithCallbackDialogData = {

type: 2,

text: this.translateService.instant('helperDialog.cancelHaveChildGoals')

}

const dialogRef = this.dialog.open(HelperWithCallbackDialogComponent, {

data

});

dialogRef.afterClosed().subscribe((result: boolean) => {

if (result) {

if (showComment) {

this.cancelWithComment(goal)

} else {

this.mainService.cancelGoal(goal.id).subscribe(() => {

this.reloadGoals(goal.kind\_id === KindEnum.kindCrossId);

})

}

}

});

}

if (showComment && !cascadedGoalExist) {

this.cancelWithComment(goal)

}

if (!showComment && !cascadedGoalExist) {

this.mainService.cancelGoal(goal.id).subscribe(() => {

this.reloadGoals(goal.kind\_id === KindEnum.kindCrossId);

});

}

});

}

Продолжение листинга 3.7

callDeleting(data: any, $event: any) {

const dataHelperWithCallback: HelperWithCallbackDialogData = {

type: 1,

text: this.translateService.instant('helperDialog.sureWantDelete')

}

const dialogRef = this.dialog.open(HelperWithCallbackDialogComponent, {

disableClose: true,

data: dataHelperWithCallback

});

dialogRef.afterClosed().subscribe((result: any) => {

if (result) {

const dataHelperWithCallback: HelperWithCallbackDialogData = {

type: 2,

text: this.translateService.instant('helperDialog.deleteHaveChildGoals')

}

const dialogRef = this.dialog.open(HelperWithCallbackDialogComponent, {

data: dataHelperWithCallback

});

dialogRef.afterClosed().subscribe((result: any) => {

if (result) {

this.mainService.deleteGoal(data.id).subscribe(() => {

this.reloadGoals();

});

}

});

}

});

}

callEditModal(data: any, $event: any) {

$event.stopPropagation();

this.createGoal(data);

}

changeShowCancels() {

sessionStorage.setItem(

'mainView\_ShowCanceled',

this.isShowCanceled.toString()

);

this.filterData()

}

changeShowGoalsWithZeroWeight() {

sessionStorage.setItem('mainView\_GoalsWithZeroWeight', this.isGoalsWithZeroWeight.toString())

this.filterData()

}

reloadGoals(refreshBaseInfo = false) {

this.mainService.getGoals(this.paramId, this.showSubGrids).subscribe((data) => {

this.goalInfo = data;

this.responceRowData = {...data};

this.filterData()

Продолжение листинга 3.7

if (refreshBaseInfo) {

this.baseInfo$.next()

}

});

}

createGoal(data?: GoalsMapEditModel) {

let user = new GoalsMapModel();

user.position\_id = this.baseInfo.position\_id;

user.position\_name = this.baseInfo.position\_name;

user.fullname = this.baseInfo.fullname;

user.subdivision\_name = this.baseInfo.division\_name;

user.is\_main\_in\_group = true;

const dataCreateEditGoalView: CreateEditGoalViewData = {

editModel: data,

map: this.paramId,

period: this.baseInfo.period\_id,

isCascading: false,

isNotEditable: false,

user: user

}

const dialogRef = this.dialog.open(CreateEditGoalViewComponent, {

disableClose: true,

width: '900px',

height: '965px',

maxHeight: '100vh',

panelClass: 'add-goal-dialog-container',

data: dataCreateEditGoalView

});

dialogRef.afterClosed().subscribe((result: boolean) => {

if (result) {

this.reloadGoals();

}

});

}

showHistory() {

const data: ViewAgreementHistoryData = {

paramId: this.paramId

}

this.dialog.open(ViewAgreementHistoryComponent, {

data,

width: '90vw',

height: '90vh',

panelClass: 'view-agreement-history-container',

});

}

toggledExpandInfo(state: boolean) {

this.expandedInfo = state;

sessionStorage.setItem(

'mainView\_expandedInfo',

this.expandedInfo.toString()

);

}

}

Данный класс имеет декоратор «@Component», поэтому представляет собой компонент Angular.

Описание полей класса MainViewComponent представлено в таблице 3.6.

Таблица 3.6 – Описание полей класса MainViewComponent

|  |  |
| --- | --- |
| Поле | Описание |
| dataGrid | Данное поле хранит в себе ссылку на таблицу |
| baseInfo | Источник данных целей карты |
| isShowCanceled | Флаг, отвечающий за отображение отмененных целей |
| goalInfo | Информация о текущей выбранной цели |
| statusesDictionary | Справочник статусов целей |

Описание методов класса MainViewComponent представлен в таблице 3.7.

Таблица 3.7 – Описание методов класса MainViewComponent

|  |  |
| --- | --- |
| Метод | Описание |
| openCascading | Метод, для отображения цели и целей, каскадированных от нее |
| addCascading | Метод, отвечающий за каскадирование выбранной цели |
| cancelWithComment | Метод для отмены цели. При отмене цели указывается комментарий, почему данная цель отменяется |
| callDeleting | Метод удаления цели |
| createGoal | Метод для создания новой цели |
| showHistory | Метод для показа истории согласования карты целей |

Продолжение таблицы 3.7

|  |  |
| --- | --- |
| changeShowCancels | Метод переключающий флаг показа отмененных целей |
| changeShowGoalsWithZeroWeight | Метод переключающий флаг показа целей с нулевым весом |

Файл HTML разметки представлен в листинге 3.8.

Листинг 3.8 – HTML разметка страницы управления картой цели сотрудника

<base-page-layout>

<header>

<div \*ngIf="baseInfo && transitionState" class="main-info-container">

<div class="main-info-row">

<div class="map-info-container">

<div class="map-info-block">

<div class="map-info-title">

{{ 'main.item.periodGoalSetting' | translate }}

</div>

<div class="map-info-value">

{{baseInfo.period\_year}}

</div>

</div>

<div class="map-info-block">

<div class="map-info-title">

{{ 'main.item.goalCardPeriod' | translate }}

</div>

<div \*ngIf="baseInfo.map\_period" class="map-info-value">

{{ baseInfo.map\_period.split(' - ')[0] | date: dateHelper.getClientFormat() }}

– {{ baseInfo.map\_period.split(' - ')[1] | date: dateHelper.getClientFormat() }}

</div>

</div>

<div class="map-info-block">

<div class="map-info-title">

{{ 'main.item.onApproval' | translate }}

</div>

<div class="map-info-value">

{{ baseInfo.assesment\_person\_name }}

</div>

</div>

<span \*ngIf="baseInfo.is\_closed === 1" class="mat-icon-grey-color">

<mat-icon

matTooltip="{{ 'mainViewHelp.lockGoalMap' | translate }}"

>lock</mat-icon>

</span>

<span \*ngIf="baseInfo.is\_closed !== 1" class="mat-icon-grey-color">

<mat-icon

matTooltip="{{ 'mainViewHelp.openGoalMap' | translate }}"

>lock\_open</mat-icon>

</span>

</div>

<div class="controls">

Продолжение листинга 3.8

<span \*ngFor="let type of transitionState | keyvalue">

<button (click)="changeGoalMapStatus(type.key)" \*ngIf="type.value" mat-button>

{{ type.key | translate }}

</button>

</span>

</div>

</div>

<div class="map-info-block buttons-map">

<div class="map-info-block">

<button (click)="navigate()" \*ngIf="goalButtonsState.allowRequest" class="add-btn withoutWidth" mat-button>

<span>{{ 'main.item.createRequestEarlyAssessmentGoalMap' | translate }}</span>

</button>

</div>

</div>

</div>

</div>

<mat-accordion \*ngIf="baseInfo">

<mat-expansion-panel (afterCollapse)="toggledExpandInfo(false)" (afterExpand)="toggledExpandInfo(true)" [expanded]="expandedInfo"

class="map-details-info">

<mat-expansion-panel-header>

<mat-panel-title>

<div class="map-details-info-container">

<div class="map-details-info-block">

<div class="map-details-info-title">

{{ 'main.item.employee' | translate }}

</div>

<div class="map-details-info-value">

{{ baseInfo.fullname }}

</div>

</div>

<div class="map-details-info-block">

<div class="map-details-info-title">

{{ 'main.item.entity' | translate }}

</div>

<div class="map-details-info-value">

{{ baseInfo.company\_name }}

</div>

</div>

<div class="map-details-info-block">

<div class="map-details-info-title">

{{ 'main.item.subdivision' | translate }}

</div>

<div class="map-details-info-value">

{{ baseInfo.division\_name }}

</div>

</div>

<div class="map-details-info-block">

<div class="map-details-info-title">

{{ 'main.item.position' | translate }}

</div>

<div class="map-details-info-value">

{{ baseInfo.position\_name }}

</div>

</div>

</div>

</mat-panel-title>

Продолжение листинга 3.8

</mat-expansion-panel-header>

</mat-accordion>

</header>

<content>

<div \*ngIf="!isMobile">

<div class="goals-list-header">

<h3>

<span>{{ 'main.item.goalList' | translate }}:</span>

<span class="helper-container" style="display: none !important;">

<!--suppress TypeScriptUnresolvedVariable -->

<app-single-helper-button-component \*ngIf="helper.tree"

[helpCode]="helper.tree.helpCode"></app-single-helper-button-component>

</span>

</h3>

</div>

<div \*ngIf="goalInfo" class="sub-info-panel-container">

<div class="sub-info-container">

<div class="sub-info-block">

<div class="sub-info-title">

{{ 'main.item.goalsCount' | translate }}

</div>

<div class="sub-info-value">

{{ goalInfo.goals ? goalInfo.goals.length : 0 }}

</div>

</div>

<div class="sub-info-block">

<div class="sub-info-title">

{{ 'main.item.weight' | translate }}

</div>

<div class="sub-info-value">

{{ goalInfo.weight | number: '1.0-0' }}

</div>

</div>

</div>

<div class="goals-grid-controls">

<dx-check-box

class="toggle-sub-grids"

(onValueChanged)="changeShowCancels()"

[(ngModel)]="isShowCanceled"

color="primary"

text="{{ 'main.item.showCancel' | translate }}"

>

</dx-check-box>

<dx-check-box

class="toggle-sub-grids"

(onValueChanged)="changeShowSubGrids($event)"

[(ngModel)]="showSubGrids"

color="primary"

text="{{ 'main.item.showCascading' | translate }}"

>

</dx-check-box>

<dx-check-box

class="toggle-sub-grids"

(onValueChanged)="changeShowGoalsWithZeroWeight()"

[(ngModel)]="isGoalsWithZeroWeight"

color="primary"

text="{{ 'main.item.goalsWithZeroWeight' | translate }}"

>

</dx-check-box>

<button (click)="createGoal()" [disabled]="!goalButtonsState.allowAddNew" class="add-btn" mat-button>

Продолжение листинга 3.8

<mat-icon>add</mat-icon>

<span>{{ 'main.item.addPersonalGoal' | translate }}</span>

</button>

</div>

</div>

<div \*ngIf="goalInfo && goalInfo.goals && goalInfo.goals.length > 0" class="map-goals-list-container">

<div

\*ngFor="let goal of goalInfo.goals; let i = index"

class="map-goals-list-item-container"

class="map-goals-list-item-container"

>

<div class="map-goals-item-first-header">

<div class="map-goal-title-container">

<span class="map-goals-number">{{i + 1}}</span>

<span \*ngIf="goal.goal\_number" class="map-goal-number">№{{ goal.goal\_number}}</span>

<span \*ngIf="!goal.goal\_number" class="map-goal-number">№-</span>

<span class="map-goal-name">{{ translateService.currentLang === 'ru' ? goal.name : goal.name\_eng }}</span>

</div>

<div class="map-goal-info-right-block">

<span \*ngIf="goal.kind\_name" class="map-goal-kind">{{ goal.kind\_name }}</span>

<span \*ngIf="goal.type\_name" class="map-goal-type">{{ goal.type\_name }}</span>

<span class="map-goal-type">{{ goal.status\_name }}</span>

<span \*ngIf="goal.source\_name" class="map-goal-source" id="{{getTooltipId(goal)}}">{{ goal.source\_name }}

</span>

<span class="map-goal-counter">

<mat-icon>attach\_file</mat-icon>

<span class="map-goal-counter-count">{{ goal.childs && goal.childs.length ? goal.childs.length : 0 }}</span>

</span>

</div>

</div>

<table class="map-goal-info-left-block">

<tr class="map-goal-info-item-block">

<th class="map-goal-info-item-title">{{ 'main.item.weight' | translate }}</th>

<th class="map-goal-info-item-title">{{ 'main.item.unit' | translate }}</th>

<th class="map-goal-info-item-title">{{ 'main.item.planAndFact' | translate }}</th>

<th class="map-goal-info-item-title">{{ 'main.item.resultSelf' | translate }}, %</th>

<th class="map-goal-info-item-title main-align-center">{{ 'main.item.resultFunctional' | translate }}, %</th>

<th class="map-goal-info-item-title main-align-center">{{ 'main.item.resultManager' | translate }}, %</th>

<th></th>

</tr>

<tr class="map-goal-info-item-block">

<td class="map-goal-info-item-value"><span class="map-goal-value-weight">{{goal.weight | displayNumber}}</span></td>

<td class="map-goal-info-item-value">{{goal.measure\_name ? goal.measure\_name : '-'}}</td>

<td class="map-goal-info-item-value"><span class="map-goal-value-plan">{{goal.result\_plan | displayNumber}}</span>

<span> / </span>

Продолжение листинга 3.8

<span class="map-goal-value-fact">{{goal.result\_fact | displayNumber}}</span></td>

<td class="map-goal-info-item-value"><div [ngClass]="{

'color-letter-e': !goal.percent\_self || +goal.percent\_self.toString() < 80,

'color-letter-d': goal.percent\_self && +goal.percent\_self.toString() >= 80 && +goal.percent\_self.toString() <= 89,

'color-letter-c': goal.percent\_self && +goal.percent\_self.toString() >= 90 && +goal.percent\_self.toString() <= 10,

'color-letter-b': goal.percent\_self && +goal.percent\_self.toString() >= 101 && +goal.percent\_self.toString() <= 120,

'color-letter-a': goal.percent\_self && +goal.percent\_self.toString() > 120

}">

<span class="map-goal-percent">{{goal.percent\_self | displayNumber}}</span>

</div></td>

<td class="map-goal-info-item-value main-align-center"><div [ngClass]="{

'color-letter-e': !goal.percent\_func\_manager || +goal.percent\_func\_manager.toString() < 80,

'color-letter-d': goal.percent\_func\_manager && +goal.percent\_func\_manager.toString() >= 80 && +goal.percent\_func\_manager.toString() <= 89,

'color-letter-c': goal.percent\_func\_manager && +goal.percent\_func\_manager.toString() >= 90 && +goal.percent\_func\_manager.toString() <= 10,

'color-letter-b': goal.percent\_func\_manager && +goal.percent\_func\_manager.toString() >= 101 && +goal.percent\_func\_manager.toString() <= 120,

'color-letter-a': goal.percent\_func\_manager && +goal.percent\_func\_manager.toString() > 120

}">

<span class="map-goal-percent">{{goal.percent\_func\_manager | displayNumber}}</span>

</div></td>

<td></td>

</tr>

</table>

<div class="map-goal-controls text-disable-selection">

<mat-icon

(click)="addCascading(goal, $event)"

\*ngIf="goalButtonsState.allowCascade"

title="{{ 'main.item.cascade' | translate }}"

>

addchart

</mat-icon>

<mat-icon

(click)="callCascading(goal, $event)"

title="{{ 'main.item.showCascade' | translate }}"

>

account\_tree

</mat-icon>

<mat-icon

(click)="callEditModal(goal, $event)"

\*ngIf="goalButtonsState.allowEdit && goal.allowEdit === 'true'"

title="{{ 'main.item.edit' | translate }}"

>

Продолжение листинга 3.8

edit

</mat-icon>

<mat-icon

(click)="callVisibilityModal(goal, $event)"

\*ngIf="!(goalButtonsState.allowEdit && goal.allowEdit === 'true')"

title="{{ 'main.item.visibility' | translate }}"

>

visibility

</mat-icon>

<mat-icon

(click)="callCanceling(goal, $event)"

\*ngIf="goalButtonsState.allowCancel && goal.allowCancel === 'true' && !goal.canceled"

title="{{ 'goalsList.cancelGoal' | translate }}"

>

close

</mat-icon>

<mat-icon

(click)="callDeleting(goal, $event)"

\*ngIf="goalButtonsState.allowDelete && goal.allowDelete === 'true'"

title="{{ 'goalsList.deleteGoal' | translate }}"

>

delete

</mat-icon>

<mat-icon

\*ngIf="!goal.showCascade || !goal.childs || !goal.childs.length"

[ngClass]="{ 'disabled-icon': !goal.childs || !goal.childs.length }"

class="toggle-icon more-icon"

title="{{ 'main.item.toggleShow' | translate }}"

>

expand\_more

</mat-icon>

<mat-icon

\*ngIf="goal.showCascade && goal.childs && goal.childs.length"

class="toggle-icon less-icon"

title="{{ 'main.item.toggleHide' | translate }}"

>

expand\_less

</mat-icon>

</div>

</div>

<div

\*ngIf="goal.childs && goal.childs.length && goal.showCascade"

class="map-goal-cascade-container"

>

<div

\*ngFor="let cascadeGoal of goal.childs"

class="map-goal-cascade-row"

>

<div class="map-goal-cascade-title">

<div class="map-goal-cascade-item-block item-name-block">

<span class="map-goal-cascade-item-value map-goal-cascade-item-number">

{{cascadeGoal.goal\_number ? cascadeGoal.goal\_number : '-'}}

</span>

<span class="map-goal-cascade-item-value">

Продолжение листинга 3.8

{{cascadeGoal.name ? cascadeGoal.name : '-'}}

</span>

</div>

<div class="map-goal-cascade-item-block">

<span class="map-goal-cascade-item-value">

{{cascadeGoal.position\_fullname ? cascadeGoal.position\_fullname : '-'}}

</span>

</div>

<div class="map-goal-cascade-item-block">

<span class="map-goal-cascade-item-title">

{{ 'main.item.goal' | translate }}:

</span>

<span

[ngClass]="{ 'status-value-red': cascadeGoal.status\_name && cascadeGoal.status\_name !== 'Действующая' }"

class="map-goal-cascade-item-value"

>

{{cascadeGoal.status\_name ? cascadeGoal.status\_name : '-'}}

</span>

</div>

<div class="map-goal-cascade-item-block">

<span class="map-goal-cascade-item-title">

{{ 'main.item.weightCascade' | translate }}:

</span>

<span class="map-goal-cascade-item-value">

{{cascadeGoal.weight | displayNumber}}

</span>

</div>

</div>

<div class="map-goal-cascade-controls">

<mat-icon

(click)="callEditModal(cascadeGoal, $event)"

\*ngIf="goalButtonsState.allowEdit && cascadeGoal.allowEdit === 'true'"

title="{{ 'main.item.edit' | translate }}"

>

edit

</mat-icon>

<mat-icon

(click)="callCanceling(cascadeGoal, $event)"

\*ngIf="goalButtonsState.allowCancel && cascadeGoal.allowCancel === 'true'"

title="{{ 'goalsList.cancelGoal' | translate }}"

>

close

</mat-icon>

<mat-icon

(click)="callDeleting(cascadeGoal, $event)"

\*ngIf="goalButtonsState.allowDelete && cascadeGoal.allowDelete === 'true'"

title="{{ 'goalsList.deleteGoal' | translate }}">

delete

</mat-icon>

</div>

</div>

</div>

<dx-tooltip

[closeOnOutsideClick]="true"

hideEvent="mouseleave"

showEvent="mouseenter"

target="#goal{{goal.id}}"

Продолжение листинга 3.8

>

<div \*dxTemplate="let data = data of 'content'">

<div class="cascad-tooltip-modal">

<h5>{{ 'helperDialog.cascading' | translate }}</h5>

<span>{{goal.parent\_position\_name}} {{goal.parent\_position\_fullname}}</span>

</div>

</div>

</dx-tooltip>

</div>

</div>

</content>

</base-page-layout>

<div \*ngIf="baseInfo" class="header-container">

<button (click)="showHistory()" class="history-btn" mat-button>

<mat-icon>history</mat-icon>

<span>{{ 'main.item.showHistory' | translate }}</span>

</button>

</div>

Разработанная страница работы с картой цели сотрудника представлена на рисунке 3.9.

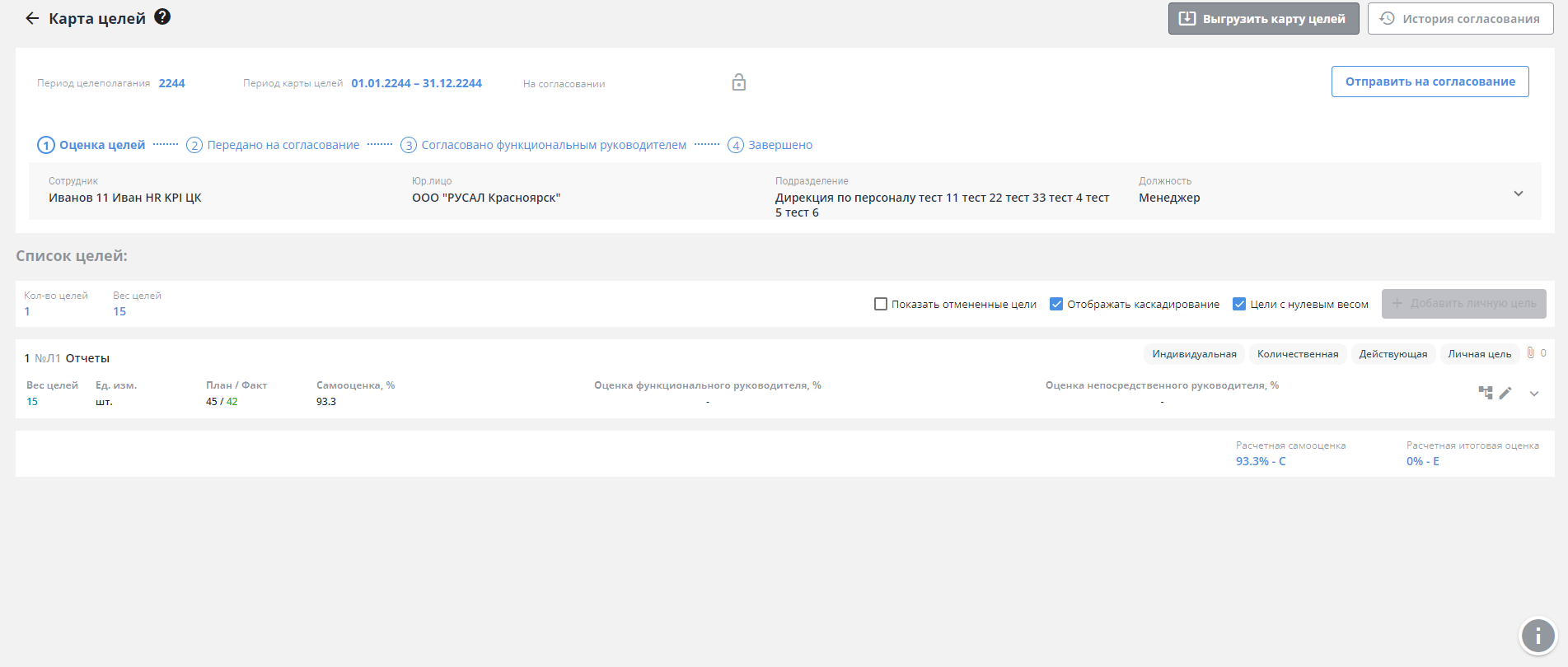


Рисунок 3.9 – Страница работы с картой цели сотрудника

На рисунке 3.10 представлено диалоговое окно редактирования цели.

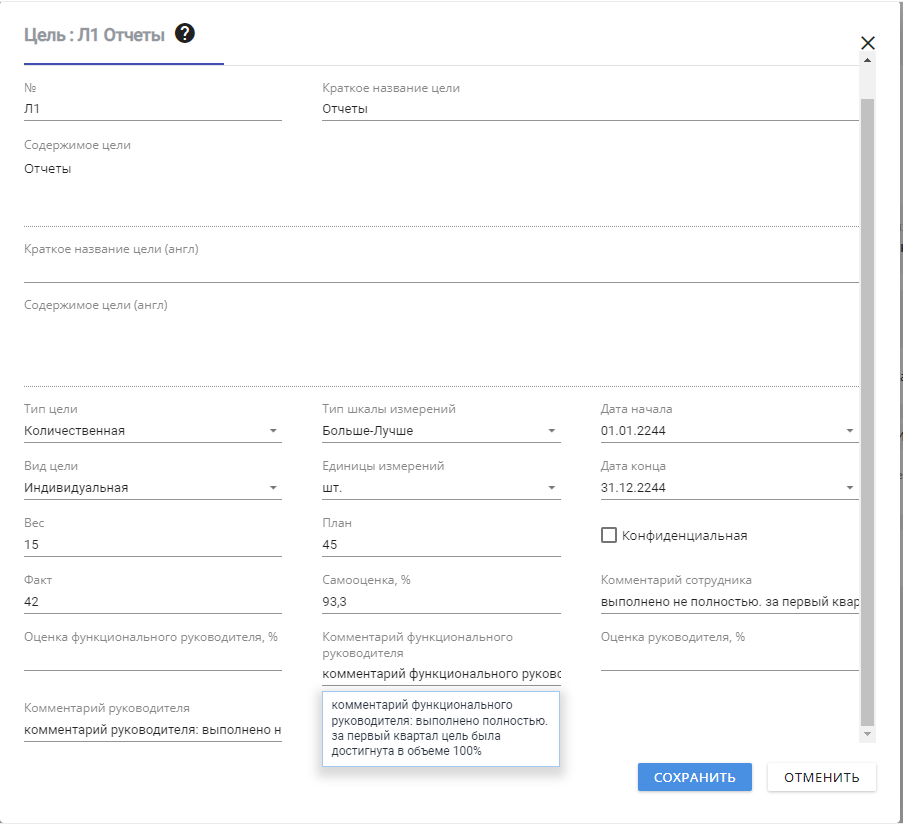


Рисунок 3.10 – Диалоговое окно добавления и редактирования цели

Таким образом, разработан Angular-компонент MainViewComponent, который необходим для управления картой цели сотрудника.

Реализация интерфейса построения отчетной документации

Логика работы интерфейса построения отчетной документации реализован несколькими классами. Каждый класс представляет собой Angular-компонент. Все компоненты схожи в реализации и отличаются только параметрами, которые необходимо заполнить для построения отчета и результирующей таблице. Каждый компонент отвечает за один отчет.

Рассмотрим класс TreeGoalsReportComponent, представленный в листинге 3.9. Данный компонент отвечает за отчет по всем целям, удовлетворяющим параметрам.

Листинг 3.9 – Класс TreeGoalsReportComponent

@Component({

selector: 'app-tree-goals',

templateUrl: './tree-goals-report.component.html',

styleUrls: ['./tree-goals-report.component.scss']

})

export class TreeGoalsReportComponent implements OnInit {

@ViewChild('reportGrid') reportGrid!: DxDataGridComponent;

orgs: OrgDictionaryModel[] = [];

model: GoalsReportFilterModel = new GoalsReportFilterModel();

subdivisions: SubdivisionDictionaryModel[] = [];

divisions: DivisionDictionaryModel[] = [];

companiesDictionary: CompanyNameModel[] = [];

report!: BaseReportModel<TreeGoalsReportModel> | null;

onOrgChange() {

this.report = null;

this.divisions = [];

this.companiesDictionary = [];

this.subdivisions = [];

this.model.division = '';

this.model.company = '';

this.model.subdivision = '';

if (this.model.org) {

this.sharedService.getDivisionsByOrgName(this.model.org).subscribe(

divisions => {

this.divisions = [...divisions];

this.subdivisions = [];

}

);

}

}

onDivisionChange() {

this.report = null;

this.model.subdivision = '';

this.subdivisions = [];

this.companiesDictionary = [];

this.model.company = ''

if (this.model.division) {

this.sharedService.getCompanyByDivision(this.model.division)

.subscribe(companies => this.companiesDictionary = companies)

}

}

onCompanyChange() {

this.report = null;

this.subdivisions = []

this.model.subdivision = ''

if (this.model.company) {

let companyName = this.companiesDictionary.find(x=>x.company\_id == this.model.company)?.company\_name??""

this.sharedService.getSubdivisionsByCompanyAndDivision(companyName, this.model.division).subscribe(

subdivisions => {

this.subdivisions = [...subdivisions]

}

);

Продолжение листинга 3.9

}

}

onChangePeriod(e: any) {

this.model.period\_id = e.value;

}

generateReport() {

this.reportTreeGoalsService.getTreeGoalsReport(this.model).subscribe(data => this.report = data)

}

resetSelectedFilters() {

this.report = null;

let period = this.model.period\_id

this.model = new GoalsReportFilterModel();

this.model.period\_id = period;

}

async exportExcelReport() {

if (this.report) {

const reportBody = new BaseReportModel<TreeGoalsReportModel>(TreeGoalsReportModel);

const filter = this.reportGrid?.instance.getCombinedFilter();

let query =Query(this.report.Rows)

if(filter){

query = query.filter(filter)

}

const rows = query.toArray();

reportBody.Form = this.report?.Form ?? {};

reportBody.Rows = rows.map((r: any) => {

delete r.Rows;

let row: TreeGoalsReportModel = new TreeGoalsReportModel()

row.org = r.org??""

row.subdivision = r.subdivision??""

row.num\_goal = r.num\_goal??""

row.num\_parent\_goal = r.num\_parent\_goal??""

row.source\_goal = r.source\_goal??""

row.confidentiality = r.confidentiality??""

row.goal\_name = r.goal\_name??""

row.goal\_content = r.goal\_content??""

row.path\_subdivision = r.path\_subdivision??""

row.view\_goal = r.view\_goal??""

row.type\_goal = r.type\_goal??""

row.rating\_scale = r.rating\_scale??""

row.measure = r.measure??""

row.goal\_result = r.goal\_result??""

row.period\_from = r.period\_from ? this.dateHelper.toClientFormat(r.period\_from) : ''

row.period\_by = r.period\_by ? this.dateHelper.toClientFormat(r.period\_by) : ''

row.responsible = r.responsible??""

row.coexecutor\_1 = r.coexecutor\_1??""

row.coexecutor\_2 = r.coexecutor\_2??""

row.coexecutor\_3 = r.coexecutor\_3??""

row.coexecutor\_4 = r.coexecutor\_4??""

row.is\_dissmised = r.is\_dissmised ? `${this.translate.instant('reports.treeGoals.is\_dissmised')}` : ''

Продолжение листинга 3.9

row.map\_status = r.map\_status ? r.map\_status : ''

row.map\_date\_from = r.map\_date\_from ? this.dateHelper.toClientFormat(r.map\_date\_from) : ''

row.map\_date\_to = r.map\_date\_to ? this.dateHelper.toClientFormat(r.map\_date\_to) : ''

return row

})

if(this.model.org){

reportBody.Form.Organization=this.orgs?.find(org=>this.model.org == org.id)?.name ?? ''

}

if(this.model.division)

{

reportBody.Form.division\_name=this.divisions?.find(division=>this.model.division == division.id)?.name ?? ''

}

if(!this.model.subdivision){

reportBody.Form.Subdivision = ""

}

this.reportTreeGoalsService.exportToExcel(reportBody).subscribe(downloadFile())

}

}

}

Описание полей класса TreeGoalsReportComponent представлено в таблице 3.8.

Таблица 3.8 – Описание полей класса TreeGoalsReportComponent

|  |  |
| --- | --- |
| Поле | Описание |
| orgs | Список организаций |
| companiesDictionary | Список юридических лиц |
| divisions | Список дивизионов |
| subdivisions | Список подразделений |
| reportGrid | Ссылка на таблицу с отчетной информацией |

Описание методов класса TreeGoalsReportComponent представлен в таблице 3.9.

Таблица 3.9 – Описание методов класса TreeGoalsReportComponent

|  |  |
| --- | --- |
| Метод | Описание |
| onOrgChange | Метод, вызывающийся при изменении параметра фильтрации «Организация». Ограничивает доступные значения параметра фильтрации «Дивизион» |
| onDivisionChange | Метод, вызывающийся при изменении параметра фильтрации «Дивизион». Ограничивает доступные значения параметра фильтрации «Юридическое лицо» |
| onCompanyChange | Метод, вызывающийся при изменении параметра фильтрации «Юридическое лицо». Ограничивает доступные значения параметра фильтрации «Подразделение» |
| generateReport | Метод, получения информации для отчета по заданным параметрам |
| resetSelectedFilters | Метод для отчистки заданных параметров |
| exportExcelReport | Метод для выгрузки построенного отчета в файл Excel |

Файл HTML разметки представлен в листинге 3.10.

Листинг 3.10 – Файл HTML разметки страницы отчета по всем целям

<base-page-layout>

<header>

<mat-card class="filters-controls position-div">

<div class="filters">

<div class="independent-filters">

<div class="widget-container w100">

<div>{{'reports.maps.periodGoalSetting' | translate}}</div>

<selectable-period-from-list

(changePeriod)="onChangePeriod($event)"

></selectable-period-from-list>

</div>

<div class="widget-container w100">

<div>{{'reports.kpi.org' | translate}}</div>

<dx-select-box (onValueChanged)="onOrgChange()" [(value)]="model.org" [items]="orgs" [searchEnabled]="true"

[showClearButton]="true"

displayExpr="name"

valueExpr="id">

</dx-select-box>

</div>

</div>

<div class="dependent-filters">

<div class="widget-container w100">

<div>{{'reports.kpi.divisionOrBusiness' | translate}}</div>

<dx-select-box (onValueChanged)="onDivisionChange()" [(value)]="model.division" [dataSource]="divisions" [searchEnabled]="true"

[showClearButton]="true"

displayExpr="name"

itemTemplate="divisionItemTemplate"

valueExpr="id"

width="300">

<div \*dxTemplate="let divisionsData of 'divisionItemTemplate'">

<dx-reports-popup

[popupText]="divisionsData.name"

[target]="divisionsData.id">

</dx-reports-popup>

</div>

</dx-select-box>

</div>

<div class="widget-container w100">

<div>{{'reports.kpi.company' | translate}}</div>

<dx-select-box (onValueChanged)="onCompanyChange()" [(value)]="model.company" [dataSource]="companiesDictionary"

[searchEnabled]="true"

[showClearButton]="true"

displayExpr="company\_name"

itemTemplate="companyItemTemplate"

valueExpr="company\_id"

width="300">

<div \*dxTemplate="let companiesDictionaryData of 'companyItemTemplate'">

<dx-reports-popup

[popupText]="companiesDictionaryData.company\_name"

[target]="companiesDictionaryData.company\_id"

>

</dx-reports-popup>

</div>

</dx-select-box>

</div>

<div class="widget-container w100">

<div>{{'reports.kpi.subdivision' | translate}}</div>

<subdivision-of-comapany-select

[subdivisions]="subdivisions"

[rootSubdivision]="model.company"

[(selected)]="model.subdivision"

></subdivision-of-comapany-select>

</div>

</div>

</div>

<div class="block-buttons">

<div class="button-container">

<dx-button

Продолжение листинга 3.10

(click)="resetSelectedFilters()"

[disabled]="!model.period\_id || !model.org"

icon="revert"

text="{{'reports.maps.resetSelectedFilters' | translate}}"

type="default"

></dx-button>

</div>

<div class="widget-container">

<dx-button

(click)="generateReport()"

[disabled]="!model.org || !model.period\_id"

text="{{'reports.maps.generateReport' | translate}}"

type="default"

></dx-button>

</div>

<div class="button-container">

<dx-button

(onClick)="exportExcelReport()"

[disabled]="!report"

text="{{'reports.maps.exportExcel' | translate}}"

type="default"

>

</dx-button>

</div>

</div>

</mat-card>

</header>

<content>

<div \*ngIf="report" class="center table-column">

<dx-data-grid

#reportGrid

[allowColumnResizing]="true"

[columnAutoWidth]="true"

[dataSource]="report.Rows"

[showBorders]="false"

class="old-design"

height="calc(100vh - 290px)"

keyExpr="org">

<dxo-paging [enabled]="false"></dxo-paging>

<dxo-scrolling columnRenderingMode="virtual"></dxo-scrolling>

<dxo-load-panel [enabled]="false"></dxo-load-panel>

<dxo-sorting mode="multiple"></dxo-sorting>

<dxo-filter-row [visible]="true"></dxo-filter-row>

<dxi-column [width]="120" caption="{{'reports.treeGoals.org' | translate}}"

dataField="org">

</dxi-column>

<dxi-column [width]="120" caption="{{'reports.treeGoals.subdivision' | translate}}"

dataField="subdivision">

</dxi-column>

<dxi-column [width]="100" caption="{{'reports.treeGoals.num\_goal' | translate}}"

dataField="num\_goal">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.num\_parent\_goal' | translate}}"

dataField="num\_parent\_goal">

</dxi-column>

Продолжение листинга 3.10

<dxi-column [width]="150" caption="{{'reports.treeGoals.goal\_name' | translate}}"

dataField="goal\_name">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.goal\_content' | translate}}"

dataField="goal\_content">

</dxi-column>

<dxi-column [width]="180" caption="{{'reports.treeGoals.path\_subdivision' | translate}}"

dataField="path\_subdivision">

</dxi-column>

<dxi-column [width]="120" caption="{{'reports.treeGoals.rating\_scale' | translate}}"

dataField="rating\_scale">

</dxi-column>

<dxi-column [width]="120" caption="{{'reports.treeGoals.measure' | translate}}"

dataField="measure">

</dxi-column>

<dxi-column [width]="180" caption="{{'reports.treeGoals.goal\_result' | translate}}"

dataField="goal\_result">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.responsible' | translate}}"

dataField="responsible">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.coexecutor\_1' | translate}}"

dataField="coexecutor\_1">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.coexecutor\_2' | translate}}"

dataField="coexecutor\_2">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.coexecutor\_3' | translate}}"

dataField="coexecutor\_3">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.coexecutor\_4' | translate}}"

dataField="coexecutor\_4">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.is\_dissmised' | translate}}"

cellTemplate="dissmisedCellTemplate"

dataField="is\_dissmised">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.map\_status' | translate}}"

dataField="map\_status">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.map\_date\_from' | translate}}"

cellTemplate="dateFromCellTemplate"

dataField="map\_date\_from">

</dxi-column>

<dxi-column [width]="150" caption="{{'reports.treeGoals.map\_date\_to' | translate}}"

cellTemplate="dateToCellTemplate"

Продолжение листинга 3.10

dataField="map\_date\_to">

</dxi-column>

<div \*dxTemplate="let data of 'dissmisedCellTemplate'">

<div>{{ !data.data.is\_dissmised ? '' : 'reports.treeGoals.is\_dissmised' | translate}}</div>

</div>

<div \*dxTemplate="let mapDate of 'dateToCellTemplate'">

<div>{{ mapDate.data.map\_date\_to ? (mapDate.data.map\_date\_to | date:dateHelper.getClientFormat()) : ''}}</div>

</div>

<div \*dxTemplate="let mapDate of 'periodFromCellTemplate'">

<div>{{ mapDate.data.period\_from ? (mapDate.data.period\_from | date:dateHelper.getClientFormat()) : ''}}</div>

</div>

<div \*dxTemplate="let mapDate of 'periodByCellTemplate'">

<div>{{ mapDate.data.period\_by ? (mapDate.data.period\_by | date:dateHelper.getClientFormat()) : ''}}</div>

</div>

</dx-data-grid>

</div>

</content>

</base-page-layout>

Разработанная страница представлена на рисунке 3.11.

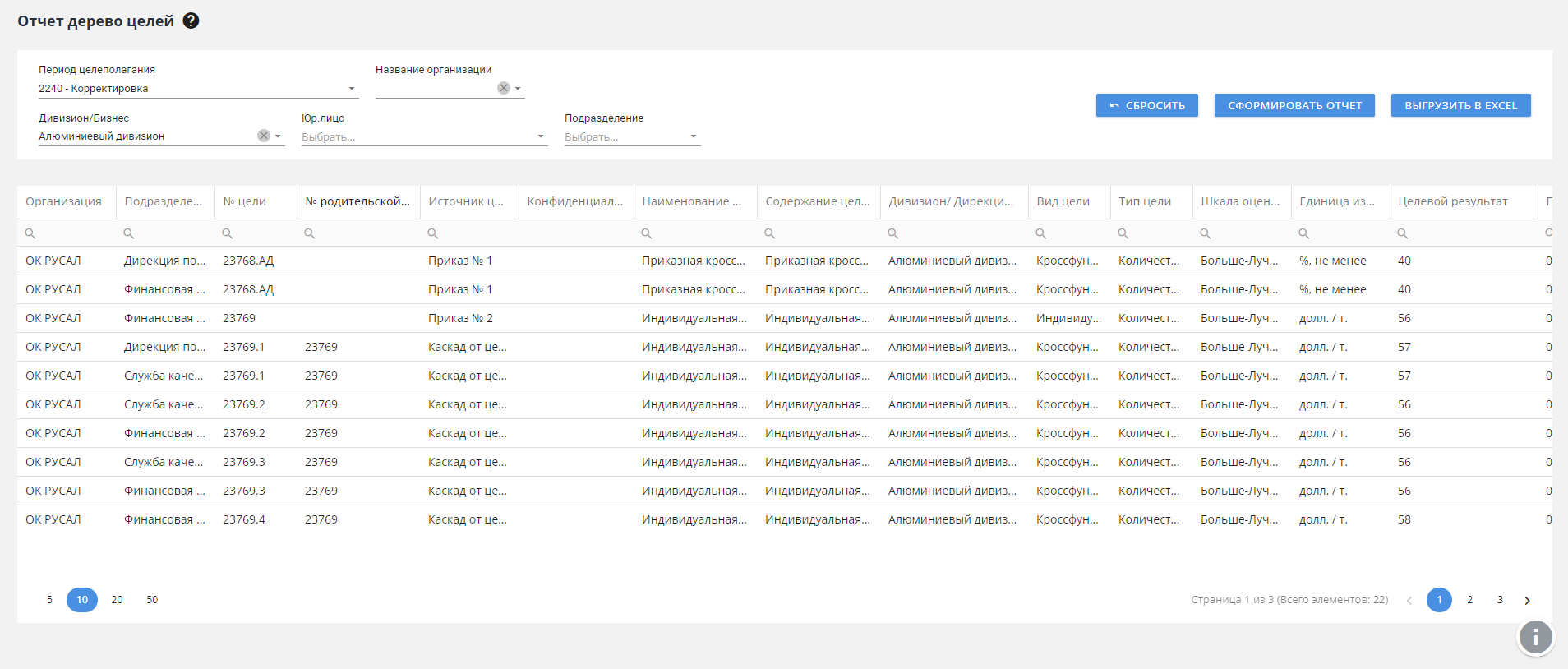


Рисунок 3.11 – Разработанная страница с отчетом

## Реализация алгоритмов

Все алгоритмы реализованы на языке программирования JavaScript.

Реализация алгоритма создания и редактирования периода целеполагания

Реализация алгоритма создания и редактирования периода целеполагания представлена в листинге 3.11.

Листинг 3.11 – Алгоритм создания и редактирования периода целеполагания

function setPeriod(){

var p = tools.read\_object(Request.Body);

if (!p.HasProperty("id")) {

throw ("wrong object format");

}

if (p.HasProperty("gs\_year") &&

p.gs\_year != "" &&

StrCharCount(p.gs\_year) == 4 &&

OptInt(p.gs\_year) != undefined)

{

var dpd = ArrayOptFirstElem(XQuery("for $elem in cc\_gs\_periods where gs\_year=" + p.gs\_year + " return $elem/Fields('id')"));

if (dpd != undefined){

if (OptInt(dpd.id) != OptInt(p.id))

throw tools\_web.get\_web\_const('kpi\_period\_exists', getCurConsts());

}}

else {

throw ("missing gs\_year");

}

if (p.HasProperty("last\_date\_employment") &&

OptDate(p.last\_date\_employment) != undefined){

if (Year(OptDate(p.last\_date\_employment)) != Int(p.gs\_year))

throw tools\_web.get\_web\_const('kpi\_admission\_end\_date\_beyond\_period\_year', getCurConsts());

}

else {

throw ("wrong object format");

}

if (p.HasProperty("last\_date\_transfer") &&

OptDate(p.last\_date\_transfer) != undefined){

if (Year(OptDate(p.last\_date\_transfer)) != Int(p.gs\_year)) {

throw tools\_web.get\_web\_const('kpi\_fransfer\_end\_date\_beyond\_period\_year', getCurConsts());

}

}

else {

throw ("wrong object format");

}

status = undefined;

if (p.HasProperty("status\_id")) {

Продолжение листинга 3.11

status = ArrayOptFirstElem(XQuery("for $elem in cc\_gs\_period\_stats where id=" + p.status\_id +" return $elem"));

if (status == undefined) {

throw ("incorrect status\_id");

}

}

else {

throw ("wrong object format");

}

pd = tools.new\_doc\_by\_name("cc\_gs\_period",false);

pd.TopElem.LoadData("<x>"+tools.object\_to\_text(p,"xml")+"</x>");

if (p.id == "")

{

//создаём

pd.BindToDb();

pd.Save();

createPeriodVersion(pd.TopElem);

}

else {

//апдейтим

before\_update\_period\_doc = tools.open\_doc(Int(p.id));

createPeriodVersion(before\_update\_period\_doc.TopElem);

pd.TopElem.date\_auto\_create = before\_update\_period\_doc.TopElem.date\_auto\_create;

pd.Save(UrlFromDocID(Int(p.id)));

//вызов агента при смене статуса

if (status != undefined) {

//агент изменения карт целей при смене статуса периода

status\_changed\_agent = ArrayOptFirstElem(XQuery("sql:

SELECT

a.id

FROM server\_agents a

WHERE a.code = 'rusal\_kpi\_period\_status\_changed'

"));

if (status\_changed\_agent == undefined) {

throw 'missing status changed agent';

}

switch(status.code) {

case '2':

hr\_kpis = XQuery("sql: SELECT");

for (hr\_kpi in hr\_kpis) {

tools.create\_notification('rusal\_kpi\_notification\_hr\_start\_goal\_setting', hr\_kpi.id, '', pd.TopElem.id);

}

pd.TopElem.date\_auto\_create = Date();

pd.Save();

tools.start\_agent(status\_changed\_agent.id, null, String(pd.TopElem.id) + "," + String(getCurUserId()));

return {

Продолжение листинга 3.11

message: tools\_web.get\_web\_const('kpi\_process\_of\_forming\_maps\_has\_started', getCurConsts())

};

break;

case '5':

hr\_kpis = XQuery("sql: SELECT ");

for (hr\_kpi in hr\_kpis) {

tools.create\_notification('rusal\_kpi\_notification\_hr\_start\_goal\_correction', hr\_kpi.id, '', pd.TopElem.id);

}

pd.TopElem.date\_auto\_create = Date();

pd.Save();

tools.start\_agent(status\_changed\_agent.id, null, String(pd.TopElem.id) + "," + String(getCurUserId()));

return {

message: tools\_web.get\_web\_const('kpi\_process\_of\_synchronizing\_has\_started', getCurConsts())

};

break;

case '3':

case '6':

case '7':

tools.start\_agent(status\_changed\_agent.id, null, String(pd.TopElem.id) + "," + String(getCurUserId()));

break;

}

}

}

return pd;

}

Первоначально в переменную p сохраняется объект, прочитанный из тела запроса. Далее идут проверки свойств объекта на валидность, а также проверка на свойство id, если оно не пустое, то происходит обновление данных о периоде, иначе создание.

Создание периода целеполагания происходит при помощи методов: new\_doc\_by\_name, LoadData, BindToDb, Save.

Обновление периода целеполагания происходит при помощи методов LoadData, Save.

Если у периода изменился статус, то вызывается серверный агент, который рассылает уведомления о смене статуса.

Реализация алгоритма создания карты целей

Реализация алгоритма создания и редактирования периода целеполагания представлена в листинге 3.12.

Листинг 3.12 – Алгоритм создания карты целей

function createGoalMap() {

request\_body = tools.read\_object(Request.Body, 'json');

position\_ids = request\_body.GetOptProperty('position\_ids');

if (position\_ids == undefined) {

throw 'missing position ids';

}

period\_id = Request.Query.GetOptProperty('period\_id');

if (period\_id == undefined) {

throw 'missing period\_id';

}

\_positions = XQuery("sql:

DECLARE @cur\_lng NVARCHAR(MAX) = '" + getCurLng() + "';

SELECT

");

if (\_positions == undefined) {

throw 'missing positions';

}

//Проверка на роль пользователя

if (!checkIfUserInGroup("kpi\_hr\_be\_") &&

!checkIfUserInGroup("kpi\_hr\_ck\_")) {

//сотрудники, для которых текущий пользователь - непосредственный руководитель

\_subs = tools\_kpi.get\_subs\_by\_boss\_manager(getCurUserId());

invalid\_positions = [];

for (\_position in \_positions) {

\_sub = ArrayOptFindByKey(\_subs, \_position.collaborator\_id, 'collaborator\_id');

if (\_sub == undefined) {

invalid\_positions.push(RValue(\_position.fullname));

}

}

if (ArrayOptFirstElem(invalid\_positions) != undefined) {

throw tools\_web.get\_web\_const('kpi\_operation\_is\_aborted\_marked\_records\_you\_have\_no\_permissions', getCurConsts()) + ": " + invalid\_positions.join(',');

}

Продолжение листинга 3.12

}

//Проверка на статус периода целеполагания

period = ArrayOptFirstElem(XQuery("sql:

SELECT

));

if (period.status\_code == '1' ||

period.status\_code == '7' ||

period.status\_code == '8' ||

period.status\_code == '9') {

throw tools\_web.get\_web\_const('kpi\_you\_may\_perform\_this\_at\_this\_period\_stage', getCurConsts());

}

invalid\_positions = [];

// Проверка на доступность добавления (есть ли на должности неотмененная карта цели)

for (\_position in \_positions) {

exists\_goalsmap = ArrayOptFirstElem(XQuery("sql:

SELECT

"));

if (exists\_goalsmap != undefined) {

invalid\_positions.push(RValue(\_position.fullname));

}

}

if (ArrayOptFirstElem(invalid\_positions) != undefined) {

throw tools\_web.get\_web\_const('kpi\_operation\_is\_aborted\_map\_has\_already\_been\_created', getCurConsts()) + ': ' + invalid\_positions.join(',');

}

invalid\_positions = [];

// Проверка на остальные должности сотрудника, если есть 3 неотмененных карт целей по другим его должностям, то выдаем сообщение с ошибкой и списком должностей

for (\_position in \_positions) {

goalsmaps = ArrayOptFirstElem(XQuery("sql:

SELECT

"));

if (OptInt(goalsmaps.count) >= 3) {

invalid\_positions.push(RValue(\_position.fullname));

}

}

if (ArrayOptFirstElem(invalid\_positions) != undefined) {

throw tools\_web.get\_web\_const('kpi\_operation\_is\_aborted\_marked\_records\_where\_employee\_already\_has\_3\_maps', getCurConsts()) + ': ' + invalid\_positions.join(',');

}

goalsmap\_active\_status = ArrayOptFirstElem(XQuery("sql:

SELECT

));

Продолжение листинга 3.12

for (\_position in \_positions) {

date\_from = Date(String(period.year) + '-01-01') > Date(\_position.position\_date) ? Date(String(period.year) + '-01-01') : Date(\_position.position\_date);

date\_to = null;

if (OptDate(\_position.position\_finish\_date) == undefined) {

date\_to = Date(String(period.year) + '-12-31');

}

else {

date\_to = Date(String(period.year) + '-12-31') > Date(\_position.position\_finish\_date) ? Date(\_position.position\_finish\_date) : Date(String(period.year) + '-12-31');

}

tools\_kpi.create\_goalsmap(\_position.position\_id, date\_from, date\_to, period\_id, goalsmap\_active\_status.id);

}

}

function create\_goalsmap( position\_id,date\_from,date\_to,period\_id,status\_id) {

new\_goal\_map\_doc = tools.new\_doc\_by\_name('cc\_gs\_goalsmap', false);

manager\_id = get\_boss\_manager(position\_id);

if (manager\_id != undefined){

new\_goal\_map\_doc.TopElem.manager\_id = manager\_id;

}

func\_manager\_id = get\_func\_manager(position\_id);

if (func\_manager\_id != undefined){

new\_goal\_map\_doc.TopElem.func\_manager\_id = func\_manager\_id;

}

goal\_map\_info = ArrayOptFirstElem(XQuery("sql:

SELECT

));

new\_goal\_map\_doc.TopElem.code = goal\_map\_info.code;

new\_goal\_map\_doc.TopElem.position\_id = position\_id;

new\_goal\_map\_doc.TopElem.date\_from = date\_from;

new\_goal\_map\_doc.TopElem.date\_to = date\_to;

new\_goal\_map\_doc.TopElem.period\_id = period\_id;

new\_goal\_map\_doc.TopElem.status\_id = status\_id;

new\_goal\_map\_doc.TopElem.is\_closed = 0;

new\_goal\_map\_doc.TopElem.is\_expired = 0;

new\_goal\_map\_doc.TopElem.date\_start\_setting = date\_from;

new\_goal\_map\_doc.BindToDb();

new\_goal\_map\_doc.Save();

goal\_map\_version = save\_map\_version(new\_goal\_map\_doc.TopElem, 1);

sync\_goals\_by\_kpis([new\_goal\_map\_doc.TopElem]);

send\_notification\_with\_subs('rusal\_kpi\_notification\_employee\_start\_goal\_setting', goal\_map\_info.collaborator\_id, '', period\_id);

return new\_goal\_map\_doc;

}

Данный алгоритм получает данные из тела запроса и проверяет их на валидность. Дальше алгоритм проверяет возможность добавления карты целей для сотрудников. Если для сотрудника невозможно создать карту целей, то алгоритм выбрасывает исключение определенного типа, в зависимости от причины недоступности создания (Неотмененная карта цели, наличие более 3 неотмененных по всем должностям сотрудника).

Непосредственное создание карты целей происходит в функции create\_goalsmap, посредством функций: new\_doc\_by\_name, BindToDb, Save.

Реализация алгоритма создания и редактирования цели

Реализация алгоритма создания и редактирования цели представлена в листинге 3.13.

Листинг 3.13 – Реализация алгоритма создания и редактирования цели

function setGoal() {

goal = tools.read\_object(Request.Body);

if (!goal.HasProperty("id")) {

throw 'wrong object format';

}

map\_id = Request.Query.GetOptProperty('map\_id');

if (map\_id == undefined) {

throw 'missing map\_id';

}

if (OptInt(map\_id) == undefined) {

throw 'wrong map\_id';

}

is\_new = OptInt(goal.id) == undefined;

if (!is\_new) {

//если цель редактируется, то проверяем,

//изменены ли поля, которые нельзя редактировать

fields\_states = getGoalFieldsStateByGoalId(goal.id);

old\_goal = ArrayOptFirstElem(XQuery("sql:

SELECT

));

if (old\_goal == undefined) {

throw 'missing old goal';

}

for (field in fields\_states) {

//если поле не редактируемое и его значение изменилось

if (fields\_states.GetOptProperty(field) == false &&

Продолжение листинга 3.13

goal.GetOptProperty(field) != undefined &&

old\_goal.GetOptProperty(field) != undefined &&

goal.GetOptProperty(field) != old\_goal.GetOptProperty(field)) {

throw tools\_web.get\_web\_const('kpi\_changed\_no\_editable\_field', getCurConsts());

}

}

}

required\_fields = [

'goal\_number',

'name',

'description',

'type\_id',

'kind\_id',

'date\_from',

'date\_to',

'coexecutors'

];

goal\_type = ArrayOptFirstElem(XQuery("sql:

SELECT

));

if (goal\_type == undefined) {

throw 'missing goal type';

}

if (goal\_type.code == 'volume') {

required\_fields = ArrayUnion(required\_fields, ['type\_scale\_id', 'measure\_id', 'result\_plan']);

if (!is\_new && fact\_is\_loaded == 1) {

required\_fields.push('result\_fact');

}

}

for (required\_field in required\_fields) {

//заполнены ли обязательные поля

if (goal.GetOptProperty(required\_field, '') == '' ||

goal.GetOptProperty(required\_field, '') == null) {

throw tools\_web.get\_web\_const('kpi\_fill\_in\_required\_boxes', getCurConsts());

}

}

if (ArrayOptFirstElem(goal.coexecutors) == undefined) {

throw tools\_web.get\_web\_const('kpi\_goal\_members\_not\_selected', getCurConsts());

}

map = undefined;

//если цель не новая, то карту берем по position\_id,

//так как для не новых целей map\_id может придти некорректный

//для каскадированных целей

if (!is\_new) {

Продолжение листинга 3.13

map = ArrayOptFirstElem(XQuery("sql:

SELECT

));

}

else {

map = ArrayOptFirstElem(XQuery("sql:

SELECT

));

}

if (map == undefined) {

throw 'missing goalsmap';

}

if (is\_new) {

new\_goal\_number = getGoalNum(0, 0, 0, goal.goal\_number);

same\_goal\_number = ArrayOptFirstElem(XQuery("sql:

SELECT

));

if (same\_goal\_number != undefined) {

throw tools\_web.get\_web\_const('kpi\_your\_map\_already\_has\_this\_goal\_number', getCurConsts());

}

goal.goal\_number = RValue(new\_goal\_number);

}

for (coexecutor in goal.coexecutors) {

position = ArrayOptFirstElem(XQuery("sql:

SELECT

));

if (position == undefined) {

throw 'coexecutor position not found';

}

coexecutor.SetProperty('position\_obj', position);

coexecutor\_map = ArrayOptFirstElem(XQuery("sql:

SELECT

));

if (coexecutor\_map == undefined) {

throw 'coexecutor map not found';

}

coexecutor.SetProperty('map\_obj', coexecutor\_map);

//создана ли для этого участника цель в его карте

is\_new\_coexecutor = ArrayOptFirstElem(XQuery("sql:

SELECT

));

if ((is\_new || is\_new\_coexecutor == undefined) &&

Продолжение листинга 3.13

OptInt(coexecutor\_map.map\_status\_code) > 8) {

incorrect\_coexecutors.push(RValue(position.basic\_collaborator\_fullname));

}

}

if (ArrayOptFirstElem(incorrect\_coexecutors) != undefined) {

throw tools\_web.get\_web\_const('kpi\_resp\_coex\_cannot\_be\_added', getCurConsts()) + ': ' + incorrect\_coexecutors.join(", ");

}

responsible = ArraySelect(goal.coexecutors, 'This.is\_main\_in\_group == 1');

if (ArrayCount(responsible) > 1) {

throw tools\_web.get\_web\_const('kpi\_can\_be\_no\_more\_one\_resp', getCurConsts());

}

else if (ArrayCount(responsible) == 0 &&

!((kind.code == 'cross' || kind.code == 'group') &&

String(goal\_source.code).indexOf('c\_') == 0)) {

throw tools\_web.get\_web\_const('kpi\_resp\_is\_not\_selected', getCurConsts());

}

responsible = ArrayOptFirstElem(responsible);

coexecutors = ArraySelect(goal.coexecutors, 'This.is\_main\_in\_group == 0');

goal\_doc = tools.new\_doc\_by\_name("cc\_gs\_goal", false);

goal\_doc.TopElem.LoadData("<x>" + tools.object\_to\_text(goal,"xml") + "</x>")

if (kind.code == 'cross' || kind.code == 'group') {

if (ArrayCount(coexecutors) == 0) {

throw tools\_web.get\_web\_const('kpi\_one\_coex\_must\_be\_specified', getCurConsts());

}

//добавление новой личной цели

if (is\_new) {

//создаем цель осн ответственного

responsible\_goal\_doc = tools.new\_doc\_by\_name("cc\_gs\_goal", false);

responsible\_goal\_doc.TopElem.LoadData("<x>" + tools.object\_to\_text(goal,"xml") + "</x>");

responsible\_goal\_doc.BindToDb();

responsible\_goal\_doc.TopElem.group\_id = responsible\_goal\_doc.TopElem.id;

responsible\_goal\_doc.TopElem.is\_main\_in\_group = 1;

if (OptDate(map.date\_from) > goal.date\_from) {

responsible\_goal\_doc.TopElem.date\_from = OptDate(StrDate(OptDate(map.date\_from), false));

}

if (OptDate(map.date\_to) < goal.date\_to) {

responsible\_goal\_doc.TopElem.date\_to = OptDate(StrDate(OptDate(map.date\_to), false));

}

responsible\_goal\_doc.Save();

//создаем цели соисполнителей

for (coexecutor in coexecutors) {

goal\_doc = tools.new\_doc\_by\_name("cc\_gs\_goal", false);

goal\_doc.TopElem.LoadData("<x>" + tools.object\_to\_text(goal,"xml") + "</x>")

Продолжение листинга 3.13

goal\_doc.TopElem.position\_id = coexecutor.position\_obj.id;

goal\_doc.TopElem.map\_id = coexecutor.map\_obj.id;

goal\_doc.TopElem.division\_id = coexecutor.position\_obj.parent\_object\_id;

goal\_doc.TopElem.group\_id = responsible\_goal\_doc.TopElem.id;

goal\_doc.TopElem.is\_main\_in\_group = 0;

if (OptDate(coexecutor.map\_obj.date\_from) > goal.date\_from) {

goal\_doc.TopElem.date\_from = OptDate(StrDate(OptDate(coexecutor.map\_obj.date\_from), false));

}

if (OptDate(coexecutor.map\_obj.date\_to) < goal.date\_to) {

goal\_doc.TopElem.date\_to = OptDate(StrDate(OptDate(coexecutor.map\_obj.date\_to), false));

}

goal\_doc.BindToDb();

goal\_doc.Save();

tools\_kpi.add\_notification\_about\_goalmap\_changes(coexecutor.position\_obj.basic\_collaborator\_id, goal\_doc.TopElem.id);

}

}

}

}

Данный алгоритм получает необходимые параметры из тела запроса и проверяет их валидность. Далее алгоритм определяет производимое действие: создание или редактирование, проверяет поля на возможность редактирования, и на заполненность обязательных полей. Если происходит редактирование цели, то обновляются поля в существующей цели, иначе создается новая. Далее алгоритм проверяет наличие соисполнителей и если они есть, то редактирует их цели или добавляет новые.

Реализация алгоритмов переключения языка

Реализация алгоритма определения языка при первом входе на портал представлена в листинге 3.14.

Листинг 3.14 – Реализация алгоритма определения языка

//На вход - строка с языком: russian, english

function gm\_set\_user\_language(language) {

resEnv.curLngID = language;

curLngWeb = lngs.GetChildByKey(language).items;

return curLngWeb;

}

//Функция для установки приоритетного языка (приоритетный язык - установленный язык в браузере)

function gm\_set\_prefer\_language() {

language\_prefer = Request.Header.GetOptProperty('Accept-Language', 'en');

language\_web = undefined;

if (language\_prefer.indexOf('ru') == 0) {

language\_web = gm\_set\_user\_language("russian");

}

else {

language\_web = gm\_set\_user\_language("english");

}

return {

'lng\_id': resEnv.curLngID,

'lng\_web': language\_web

};

}

Функция gm\_set\_prefer\_language получает в переменную language\_prefer язык, установленный в браузере пользователя. Если получить язык не удалось, то по умолчанию используется английский язык. В глобальную переменную системы resEnv.curLngID устанавливается полученный язык.

Реализация алгоритма переключения языка представлена в листинге 3.15.

Листинг 3.15 – Алгоритм переключения языка

sAction = Request.Query.GetOptProperty( 'action' , 'set\_language' );

switch (sAction)

{

    case 'set\_language':

        Env.curLngSelectedByUser = true;

        if (resEnv.curLngID == "english")

        {

            Request.AddRespHeader( 'Set-Cookie', 'cur\_language=ru' + ';' + ( AppConfig.HasProperty( 'SESSION-COOKIE-SECURE' ) == '1' ? ';Secure' : '' ));

            resEnv.curLngID = "russian";

        }

        else if (resEnv.curLngID == "russian")

Продолжение листинга 3.15

        {

            Request.AddRespHeader( 'Set-Cookie', 'cur\_language=en' + ';' + ( AppConfig.HasProperty( 'SESSION-COOKIE-SECURE' ) == '1' ? ';Secure' : '' ));

            resEnv.curLngID = "english";

        }

        break;

}

Данный алгоритм проверяет текущее значение глобальной переменной resEnv.curLngID, если оно равно «english», то устанавливается русский язык, иначе английский.

Реализация алгоритма построения отчетной документации

Все алгоритмы построения отчетной документации схожи в реализации. Рассмотрим функцию getGoalTreeReport, представленную в листинге 3.16.

Данная функция отвечает за выборку данных для отчета по всем целям, удовлетворяющим заданным параметрам.

Листинг 3.16 – Функция getGoalTreeReport

function getGoalTreeReport() {

period\_id = Request.Query.GetOptProperty('period\_id');

if (period\_id == undefined) {

throw 'missing period\_id';

}

if (OptInt(period\_id) == undefined) {

throw 'wrong format period\_id';

}

company\_name = Request.Query.GetOptProperty('company\_name', '');

org\_name = Request.Query.GetOptProperty('org\_name', '');

division\_id = Request.Query.GetOptProperty('division\_id', '');

subdivision\_id = Request.Query.GetOptProperty('subdivision\_id', '');

//Пользователь, формирующий отчеты, должен иметь роль «HR-администратор KPI»

cur\_user\_id = getCurUserId();

is\_admin = (getCurUserAccessRole() == "admin") ? 1 : 0

subdiv\_ids\_str = "0"

title = tools\_web.get\_web\_const('kpi\_report\_goals\_tree', getCurConsts());

return ArrayOptFirstElem(XQuery("sql:

DECLARE @period\_id BIGINT = " + period\_id + ",

@subdivisions\_ids NVARCHAR(MAX) = '" + subdiv\_ids\_str + "',

Продолжение листинга 3.16

@is\_admin BIT = " + is\_admin + ",

@cur\_user\_id BIGINT = " + cur\_user\_id + ",

@title NVARCHAR(max) = '" + title + "',

@cur\_lng NVARCHAR(max) = '" + getCurLng() + "',

--фильтры

" + query\_variables + "

SELECT

(

SELECT

...

FOR JSON PATH, INCLUDE\_NULL\_VALUES

), '[]') 'Rows'

FOR JSON PATH, WITHOUT\_ARRAY\_WRAPPER

) data;

")).data;

}

Данная функция получает необходимые параметры из запроса, проверяет роль пользователя на значение «HR-администратор KPI», формирует заголовок отчета, далее по полученным параметрам производится выборка из базы данных и возвращается из функции в виде JSON-файла[8].

## Выводы по разделу

Таким образом, в данном разделе решены следующие задачи:

* настроено окружение разработки: установлены необходимые зависимости для разработки, настроено форматирование кода при помощи Prettier;
* реализованы пользовательские интерфейсы в виде Angular-компонентов;
* реализованы алгоритмы работы программного обеспечения на языке программирования JavaScript.

# Обеспечение информационной безопасности

## Определение класса защищенности информационной системы и уровня значимости информации

Согласно приказу ФСТЭК России от 11 февраля 2013 года организационные и технические меры защиты информации, реализуемые в рамках системы защиты информации информационной системы должны быть направлены на исключение:

* обеспечение конфиденциальности информации;
* обеспечение целостности информации;
* обеспечение доступности информации;

Информационная система имеет федеральный масштаб, так как она функционирует на территории нескольких субъектов Российской Федерации и за ее пределами.

Для определения уровня значимости информации используют степень возможного ущерба для обладателя информации.

Информация имеет высокий уровень значимости, если степень возможного ущерба для одного из свойств обеспечения безопасности (конфиденциальность, целостность, доступность) высокая. Информация имеет средний уровень значимости, если степень возможного ущерба для одного из свойств обеспечения безопасности средняя. Информация имеет низкий уровень значимости, если степень возможного ущерба для свойств не является высокой или средней.

Соответствие свойств безопасности и степеней ущерба представлено в таблице 4.1.

Таблица 4.1 – Соответствие свойств безопасности и степеней ущерба

|  |  |  |
| --- | --- | --- |
| Свойство безопасности | Степень ущерба | Описание |
| Конфиденциальность | Высокая | Неправомерное копирование, предоставление или распространение информации может привести к серьезным негативным последствиям, так как информационная система содержит личные данные сотрудников: ФИО, номер телефона, почта, паспортные данные |
| Целостность | Высокая | Неправомерное уничтожение или модифицирование информации может привести к существенным негативным последствиям: потере информации о сотрудниках и их деятельности, ошибкам в расчетах ключевых показателей производительности сотрудников и подразделений |
| Доступность | Средняя | Неправомерное блокирование информации может привести к негативным последствиям, информационная система не сможет функционировать некоторое время, но после восстановления доступа, сотрудники и их руководители смогут зафиксировать всю работу, которую они сделали за время отказа системы и система продолжит функционировать в привычном режиме. |

Таким образом, информация имеет высокий уровень значимости, так как степень ущерба для нескольких свойств имеет высокую степень ущерба.

Устанавливаются три класса защищенности информационной системы, определяющие уровни защищенности содержащейся в ней информации:

* К1 – Первый класс, наиболее высокий
* К2 – Второй класс
* К3 – Третий класс, наиболее низкий

Также для системы устанавливается класс защищенности к информации, зависящий от ее масштаба и уровня значимости информации.

Класс защищенности информационной системы определяется в соответствии с таблицей 4.2.

Таблица 4.2 – Классы защищенности информационной системы

|  |  |  |  |
| --- | --- | --- | --- |
| Уровень значимости информации | Масштаб информационной системы | | |
| Федеральный | Региональный | Объектовый |
| Высокий | К1 | К1 | К1 |
| Средний | К1 | К2 | К2 |
| Низкий | К2 | К3 | К3 |

Таким образом, программная система имеет первый класс защищенности информационной системы, так как уровень значимости информации системы – высокий, а масштаб системы – федеральный.

## Примененные меры обеспечения информационной безопасности

Для аутентификации пользователей в системе используется комбинация логина и пароля. Портал работает на основе сетевого протокола HTTPS. Протокол HTTPS является расширением протокола HTTP для поддержки шифрования в целях повышения безопасности. Данный протокол обеспечивает защиту от атак, основанных на прослушивании сетевого соединения.

Также доступ к порталу имеют только сотрудники, учетные записи которых зарегистрированы в домене компании, а сам портал функционирует в локальной сети, что повышает защиту от несанкционированного доступа. Пароли сотрудников хэшируются с использованием криптографической односторонней хэш-функции SHA-256.

Доступ к функционалу системы разграничен по ролям.

# Технико-экономическое обоснование

## Расчет затрат на разработку

Целью экономической части является расчет эффективности от внедрения программного обеспечения задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM.

На разработку программной платформы компанией-разработчиком выделено 1 специалист на 8 месяцев работы. Ежемесячная заработная плата выделенного сотрудника составляет 45 тысяч рублей. Однако, заработная плата облагается государственными налогами и отчислениями: налог на доход физический лиц (НДФЛ), взносы в Пенсионный фонд, в Фонд социального страхования, в Федеральный фонд обязательного медицинского страхования (ФОМС), взносы на страхование от несчастных случаев и профессиональных заболеваний. Общий размер налогов и отчислений составляет 43,2%.

Таким образом, затраты на выплату заработной платы сотруднику за восемь месяцев работы составляют:

8 \* 45 000 \* 1,432 = 515 520 рублей

Помимо затрат на заработную плату сотруднику существуют еще расходы на ввод в эксплуатацию программного обеспечения. Для ввода в эксплуатацию требуется выделенный сервер, а также наличие лицензии на программную платформу Websoft HCM. Но компания-заказчик имеет выделенный сервер с установленной платформой, поэтому дополнительных затрат на ввод в эксплуатацию не требуется.

## Расчет экономического эффекта

Учитывая, что в ходе эксплуатации программного обеспечения могут возникнуть различные ошибки, дополнительно предусмотрено 300 000 рублей на сопровождение продукта в первый год эксплуатации.

Экономический эффект от программной системы достигается за счет экономии времени работы менеджеров компании по контролю за деятельностью сотрудников, времени для сбора данных для отчетной документации, а также повышении показателей производительности сотрудников.

Ожидаемое сокращение расходов в первый год эксплуатации составляет от 3 000 000 до 7 000 000 рублей. Предполагаемый срок окупаемости реализованного программного обеспечения составит:

(515 520 + 300 000) / 3 000 000 / 12 = 3, 26 месяца

(515 520 + 300 000) / 3 000 000 / 12 = 1, 39 месяца

Таким образом, предполагаемый срок окупаемости программного обеспечения составляет от 1,39 месяца до 3, 26 месяцев.

Стоит отметить, что данное программное обеспечение является универсальным, не привязанным к конкретной компании. Таким образом, программное обеспечение может быть реализовано и оптимизировано под разных заказчиков, что принесет дополнительную прибыль.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы спроектировано и разработано программное обеспечение задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM.

Для достижения поставленной цели решены следующие задачи:

* изучены и проанализированы концепция управления предприятием по целям, сбалансированная система показателей и ключевые показатели эффективности;
* разработаны бизнес-требования и функциональные требования к программному обеспечению для оценки и анализа ключевых показателей производительности работы сотрудников предприятия;
* исследовано устройство программной платформы Websoft HCM и ее функциональные возможности;
* спроектированы бизнес-процесс оценки и анализа ключевых показателей производительности работы сотрудников, схема базы данных и система разграничения доступа к информации;
* выбраны средства реализации программного обеспечения и настроено окружение разработки;
* спроектированы алгоритмы работы программного обеспечения, а также пользовательские интерфейсы для работы с ним;
* разработана система разграничения доступа к информации;
* разработаны алгоритмы работы программного обеспечения;
* разработаны пользовательские интерфейсы;
* определен класс защищенности информационной системы и уровень значимости информации, описаны применяемые меры обеспечения информационной безопасности.

Таким образом, цель выпускной квалификационной работы достигнута; задачи выпускной квалификационной работы решены в полном объеме. Целесообразность разработанного программного обеспечения доказана технико-экономическими расчетами.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Афанасьева Л. А. Методологические аспекты управления мотивацией и стимулированием трудовой деятельности компании // Основы экономики, управления и права. – 2013. – № 5 (11) [Электронный ресурс]. – URL: http://cyberleninka.ru/article/n/metodologicheskie-aspekty-upravleniya-motivatsiey-i-stimulirovaniem-trudovoy-deyatelnosti-kompanii.
2. Мотивация персонала: учеб. пособие (практикум) / Ю. Г. Одегов, Г. Г. Руденко, С. Н. Апенько, А. И. Мерко. – М. : Альфа-Пресс, 2010. – 633 с.
3. Клиентский портал Websoft [Электронный ресурс] URL: https://news.websoft.ru/\_wt/default
4. The C4 model for visualising software architecture [Электронный ресурс] URL: Level 3: Component diagram <https://c4model.com/>
5. Angular [Электронный ресурс] URL: <https://angular.io/>
6. Советов Б.Я. Базы данных: теория и практика: учебник для бакалавров / Б.Я. Советов, В.В. Цехановский, В.Д. Чертовский. – 2–е изд. – М.: Издательство Юрайт, 2014. – 463 с.
7. DevExtreme - JavaScript UI Components for Angular, React, Vue and jQuery by DevExpress [Электронный ресурс] URL: <https://js.devexpress.com/>
8. Microsoft. SQL Docs [Электронный ресурс] URL: SQL Server technical documentation https://docs.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver16

ПРИЛОЖЕНИЕ А

**Техническое задание**

**Программное обеспечение задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM**

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**на 5 страницах**

СОДЕРЖАНИЕ

[[1 Общие сведения 112](#_Toc105372798)](#_Toc106022768)

[[1.1 Полное наименование системы и ее условное обозначение 112](#_Toc105372798)](#_Toc106022769)

[[2 Назначение и цели создания системы 112](#_Toc105372798)](#_Toc106022770)

[[2.1 Назначение системы 112](#_Toc105372798)](#_Toc106022771)

[[2.2 Цель и задачи создания системы 112](#_Toc105372798)](#_Toc106022772)

[[3 Характеристика объектов автоматизации 112](#_Toc105372798)](#_Toc106022773)

[[3.1 Краткие сведения об объекте автоматизации 112](#_Toc105372798)](#_Toc106022774)

[[4 Требования к системе 112](#_Toc105372798)](#_Toc106022775)

[[4.1 Требования к системе в целом 112](#_Toc105372798)](#_Toc106022776)

[[4.1.1 Требования к характеристикам взаимосвязей создаваемой системы со смежными системами, требования к ее совместимости 112](#_Toc105372798)](#_Toc106022777)

[[4.1.2 Требования к квалификации персонала, порядку его подготовки и контроля знаний и навыков 113](#_Toc105372798)](#_Toc106022778)

[[4.1.3 Требования к безопасности 113](#_Toc105372798)](#_Toc106022779)

[[4.1.4 Требования к системе, связанные с особыми условиями эксплуатации 113](#_Toc105372798)](#_Toc106022780)

[[4.1.5 Специальные требования по усмотрению разработчика или заказчика системы 113](#_Toc105372798)](#_Toc106022781)

[[4.1.6 Требования к функциям (задачам), выполняемым системой 113](#_Toc105372798)](#_Toc106022782)

1. **Общие сведения**
   1. Полное наименование системы и ее условное обозначение

Полное наименование – «Программное обеспечение задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM».

1. Назначение и цели создания системы
   1. Назначение системы

Автоматизация процесса оценки и анализа ключевых показателей производительности работы сотрудников предприятия

* 1. Цель и задачи создания системы

Целью является проектирование и разработка программного обеспечения задачи оценки и анализа ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM.

1. Характеристика объектов автоматизации
   1. Краткие сведения об объекте автоматизации

Объектом автоматизации является оценка и анализ ключевых показателей производительности работы сотрудников предприятия на базе платформы Websoft HCM.

1. Требования к системе
   1. Требования к системе в целом

Общие требования к программе:

* выполнение всех функций, предусмотренных настоящим документом в пункте 4.1.6;
  + 1. Требования к характеристикам взаимосвязей создаваемой системы со смежными системами, требования к ее совместимости

Не предъявляются.

* + 1. Требования к квалификации персонала, порядку его подготовки и контроля знаний и навыков

Программная система предназначена для использования всеми сотрудниками компании. Специальная подготовка и навыки не требуется.

* + 1. Требования к безопасности

Программная система должна отвечать требованиям о защите информации ФСТЭК согласно ее классу защищенности.

* + 1. Требования к системе, связанные с особыми условиями эксплуатации

Программная система должна встраиваться в программную платформу Websoft HCM.

* + 1. Специальные требования по усмотрению разработчика или заказчика системы

Заказчик предоставляет разработчику доступ к серверу с предустановленной и сконфигурированной СУБД Microsoft SQL Server, а также с предустановленной и сконфигурированной программной платформой Websoft HCM.

* + 1. Требования к функциям (задачам), выполняемым системой

˗ Встраивание ПО в программную платформу Websoft HCM;

˗ Автоматическое разграничение функционала по ролям доступа;

˗ Система автоматически контролирует периоды целедостижения;

˗ Автоматическое формирование карт целей для каждого сотрудника;

˗ Разработка количественных и качественных показателей для измерения степени достижения целей;

˗ Автоматическое выстраивание иерархии целей;

˗ Обеспечение связи между сотрудником и руководством: через системы рассылки уведомлений и комментирования целей;

˗ Разработка системы для личных целей сотрудников;

˗ Обеспечение обратной связи между сотрудниками компании;

˗ Разработка автоматического построения отчетной документации;

* Поддержка мультиязычности.